TESIS TE142599

SISTEM AUTO DOCKING PADA SERVICE ROBOT MENGGUNAKAN PERSEPSI VISUAL

RIZA AGUNG FIRMANSYAH
NRP. 2212204014

DOSEN PEMBIMBING
Ir. Djoko Purwanto, M.Eng., Ph.D.
Ronny Mardiyanto, S.T., M.T., Ph.D.

PROGRAM MAGISTER
BIDANG KEAHLIAN TEKNIK ELEKTRONIKA
JURUSAN TEKNIK ELEKTRO
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
2015

THESIS TE142599

AUTO DOCKING SYSTEM FOR SERVICE ROBOT USING VISUAL PERCEPTION

RIZA AGUNG FIRMANSYAH
2212204014

SUPERVISOR
Ir. Djoko Purwanto, M.Eng., Ph.D.
Ronny Mardiyanto, S.T., M.T., Ph.D.

MAGISTER PROGRAM
FIELD IN ELECTRONICS
ELECTRICAL DEPARTMENT
FACULTY OF INDUSTRIAL TECHNOLOGY
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
2015

Tesis ini disusun untuk memenuhi salah satu syarat memperoleh gelar
Magister Teknik (MT)
di
Institut Teknologi Sepuluh Nopember

Oleh
RIZA AGUNG FIRMANSYAH
NRP. 2212204014

Tanggal Ujian : 8 Januari 2015
Periode Wisuda : Maret 2015

Disetujui oleh :

Dosen Penguji :

1. **Achmad Arifin, ST., M.Eng., Ph.D.**
   NIP. 19710314 199702 1 001

2. **Dr. Muhammad Rivai, ST., MT.**
   NIP. 19690426 199403 1 003

3. **Dr. Tri Arief Sardjono, ST., MT.**
   NIP. 19700212 199512 1 001

Dosen Pembimbing:

1. **Ir. Djoko Purwanto, M.Eng., Ph.D .**
   NIP. 19651211 199002 1 002

2. **Ronny Mardiyanto, ST., MT., Ph.D.**
   NIP. 19810118 200312 1 003

Direktur Program Pascasarjana,

Prof. Dr. Ir. Adi Soeprijanto, MT.
NIP. 19640405 199002 1 001

SISTEM *AUTO DOCKING* PADA *SERVICE ROBOT* MENGGUNAKAN PERSEPSI VISUAL

Nama Mahasiswa	: Riza Agung Firmansyah
NRP	: 2212204014
Pembimbing	: 1.Ir. Djoko Purwanto, M.Eng., Ph.D
	2. Ronny Mardiyanto, ST., MT., Ph.D

ABSTRAK

Sistem *auto docking* adalah sebuah sistem pada sebuah robot yang berfungsi untuk melakukan pengisian baterai secara otomatis. Sistem ini dijalankan saat robot mendeteksi tegangan kerja minimal. Robot melakukan pengisian baterai pada sebuah *docking station*. Sistem *auto docking* ini dibangun dengan menggunakan persepsi visual berbasis *local binary pattern* (LBP) *histogram matching* agar robot mampu mendeteksi *docking station*. Setelah pendeteksian berhasil, dilanjutkan dengan ekstraksi fitur untuk mendapatkan posisi dan orientasi *docking station*. Selanjutnya posisi dan orientasi dijadikan sebagai informasi masukan *fuzzy logic controller* untuk menjalankan robot. Robot menjalankan sistem *auto docking* menggunakan persepsi visual saat tegangan baterai dibawah 23.6 volt dan menghentikan pengisian baterai saat tegangan baterai diatas 26.4 volt. Pengisian baterai dilakukan dalam waktu 135 menit. Sistem *auto docking* mampu bekerja dengan tingkat akurasi 86.7 % dan optimal pada rentang luminasi 116 lux hingga 395 lux. Robot melakukan penyambungan konektor dengan rata-rata waktu 53.78 detik dari jarak 450 cm dengan akurasi penyambungan konektor mencapai 84%.

Kata kunci: *service robot*, *auto docking*, persepsi visual, *local binary pattern*, *histogram matching, fuzzy logic controller.*

Halaman ini sengaja dikosongkan

AUTO DOCKING SYSTEM FOR *SERVICE ROBOT* USING VISUAL PERCEPTION

Name	: Riza Agung Firmansyah
NRP	: 2212204014
Supervisor	: 1.Ir. Djoko Purwanto, M.Eng., Ph.D
	2. Ronny Mardiyanto, ST., MT., Ph.D

ABSTRACT

Auto docking system is a system on a robot that has a function to make the battery charging automatically. This system is executed when the robot detects a minimum working voltage. Robot perform battery charging on a docking station. Auto docking system was built using visual perception based on local binary pattern (LBP) histogram matching to detect a docking station. After docking station detected, then feature extraction is performed to get the posistion and orientation of the docking station. Position and orientation is used as fuzzy logic controller input to move the robot. Robot executed the auto docking system using visual perception when the battery voltage is below 23.6 volts and finish charging the battery when voltage is over 26.4 volts. Battery charging is done within 135 minutes. Auto docking system using visual perception has an accuration 86.7% and optimally work at luminance 116 lux up to 395 lux. The robot is able to perform splicing connector with an average time of 53.78 seconds from a distance of 450 cm with an accuracy of 84%.

Keywords: service robot, auto docking, visual perception, local binary pattern, histogram matching, fuzzy logic controller.

Halaman ini sengaja dikosongkan

KATA PENGANTAR

Alhamdulillah, puji dan syukur penulis panjatkan kepada Allah SWT, karena atas segala nikmat-Nya lah tesis ini dapat diselesaikan. Tesis berjudul "Sistem *Auto docking* pada *Service robot* Menggunakan Persepsi Visual" ini disusun untuk memenuhi sebagian persyaratan memperoleh gelar Magister Teknik (M.T) pada Jurusan Teknik Elektro, Fakultas Teknologi Industri, Institut Teknologi Sepuluh Nopember.

Penulis menyadari bahwa dalam penyusunan tesis ini tidak terlepas dari bantuan berbagai pihak. Oleh karena itu, dengan ketulusan dan kerendahan hati penulis menyampaikan terima kasih kepada:

1. Untuk kedua orang tuaku tercinta Bapak Sukartono dan Ibu Sini yang selalu memberi dukungan dan mendo'akan saya .
2. Bapak Ir. Djoko Purwanto, M.Eng., Ph.D. dan Bapak Ronny Mardiyanto, S.T, M.T, Ph.D. selaku dosen pembimbing yang telah banyak memberikan saran, bantuan, serta bimbingan.
3. Bapak Dr. Tri Arief Sardjono, ST., MT., selaku Ketua Jurusan Teknik Elektro Institut Teknologi Sepuluh Nopember Surabaya.
4. Bapak Achmad Arifin, ST., MT., Ph.D. selaku koordinator Bidang Studi Elektronika.
5. Seluruh dosen pengajar Jurusan Teknik Elektro yang telah banyak memberikan ilmu selama penulis menempuh kuliah.
6. Pusat Robotika ITS dan seluruh staf atas segala fasilitas dan bantuannya.
7. Rekan–rekan seperjuangan S2 Elektronika ITS atas segala bantuan dan sumbangan pikiran dan tenaga dalam menyelesaikan tesis ini.
8. Semua kakak-kakak ku Mbak Erwin dan Mas Wimar, Mbak Ratna dan Mas Yudi, dan Juga bu lik Sumarmi dan keponakan Alifia "Fira" Febrianti
9. Semua pihak yang tidak mungkin bagi penulis untuk mencantumkan satu-persatu, terimakasih banyak atas bantuan dan dukungannya.

Pada akhirnya, penulis menyadari bahwa tesis ini masih belum sempurna. Oleh karena itu, penulis sangat mengharapkan kritik dan saran yang membangun. Penulis berharap semoga tesis ini dapat bermanfaat bagi pengembangan ilmu pengetahuan dan teknologi serta bagi masyarakat.

Surabaya, Januari 2015

Penulis

DAFTAR ISI

DAFTAR GAMBAR

Halaman ini sengaja dikosongkan

DAFTAR TABEL

Halaman ini sengaja dikosongkan

BAB 1

PENDAHULUAN

1.1 Latar Belakang

Dalam beberapa tahun terakhir, aplikasi robot sudah banyak dikembangkan pada beberapa bidang seperti pada militer, industri, pendidikan, kesehatan dan lain-lain. Sehingga tidak menutup kemungkinan jika aplikasi interaksi manusia dan robot akan semakin berkembang [1]. Aplikasi berkembangnya teknologi robot salah satunya adalah aplikasi robot pada rumah tangga seperti *service robot* atau robot pembantu.

Dalam melakukan pekerjaanya, tidak menutup kemungkinan jika sumber energi yang digunakan robot akan habis sebelum semua pekerjaanya selesai. Sehingga sumber energi robot harus segera diisi ulang karena jika tidak robot akan berhenti berfungsi atau mati akibat sumber energinya habis. Maka untuk mencegah hal tersebut robot harus mampu memperkirakan atau membaca sisa energi pada sumber energinya dan jika mendekati suatu nilai ambang batas tertentu robot harus menghentikan pekerjaanya dan melakukan proses pengisian baterai [2].

Sistem pengisian secara otomatis juga pernah diteliti oleh [3] yang telah menerapkan sistem *auto docking* pada mobile robot menggunakan sensor ultrasonic. Sistem ultrasonic memiliki kelemahan dimana lokasi *docking* harus ditentukan dan posisinya tetap karena tidak memiliki sistem secara visual. Kemudian [4] menggunakan sebuah infrared *rangefinder* untuk masukan robot dalam menemukan *docking*. Sistem tersebut memiliki kelemahan yang sama dengan sistem yang menggunakan ultrasonic.

Sistem yang dibangun dengan *infrared* maupun ultrasonic tidak mampu bekerja saat lokasi *docking station* diubah. Maka untuk memperbaiki kekurangan dalam pengenalan lokasi *docking* maka digunakan persepsi visual. Sistem *auto docking* menggunakan persepsi visual telah dilakukan oleh [5] dan [6]. Mereka

menggunakan sebuah kamera untuk melakukan *docking* secara otomatis. Kamera tersebut digunakan untuk mendeteksi sebuah penanda yang menunjukan *docking station*.

Persepsi visual memiliki kendala perubahan luminasi cahaya di lingkungan kerjanya. Sistem pendeteksi obyek yang berbasis intensitas warna akan sangat terganggu oleh masalah tersebut. Pendeteksi obyek yang baik harus mampu mengatasi permasalahan tersebut. Sehingga sistem pendeteksi obyek harus dibuat berdasarkan pola, bentuk maupun fitur-fitur yang berada dalam obyek yang dicari. Pola, bentuk maupun fitur lebih tahan terhadap perubahan luminasi dibanding dengan pendeteksi obyek dengan warna.

Dengan dasar-dasar permasalahan tersebut maka sistem *auto docking* dengan persepsi visual yang akan dikembangkan dalam penelitian ini harus tahan terhadap luminasi cahaya. Sistem yang dibuat diharapkan mampu bekerja pada rentang luminasi yang lebih banyak walaupun tetap mengalami kesulitan pada luminasi ekstrim. Persepsi visual pada sistem *auto docking* ini dibangun untuk menemukan *docking station*. Setelah menemukan *docking station* robot akan bergerak menuju *docking station* dan kemudian menghubungkan konektor ke *charger*.

1.2 Perumusan Masalah

Dalam melakukan penelitian sistem *auto docking* pada *service robot* menggunakan persepsi visual, terdapat beberapa permasalahan yang akan diangkat. Agar sistem *auto docking* dapat berjalan sesuai keinginan, maka beberapa proses mengenai persepsi visual harus dapat dilakukan dengan baik.

Permasalahan yang akan diselesaikan meliputi:

1. Penentuan waktu yang tepat untuk menjalankan sistem *auto docking* akan mencegah robot kehabisan energi baterai sebelum melakukan pengisian baterai.
2. Lokasi *docking station* yang belum diketahui dengan kondisi luminasi cahaya yang berubah-ubah akan menyulitkan robot menemukan *docking station* dengan menggunakan persepsi visual.

3. Setelah *docking station* ditemukan robot harus bergerak menuju *docking station* dan menyambungkan konektor pada *charger*. Namun untuk menggerakan robot membutuhkan sebuah kontroler yang mampu bekerja berdasarkan persepsi visual.

1.3 Tujuan dan Manfaat Penelitian

Penelitian sistem *auto docking* pada *service robot* menggunakan persepsi visual memiliki beberapa tujuan. Dengan tercapainya tujuan tersebut diharapkan semua permasalahan yang terjadi dapat diselesaikan. Tujuan dari penelitian meliputi :

1. Membuat sistem pembaca tegangan baterai untuk mengetahui kapan sistem *auto docking* dijalankan.
2. Membuat sistem pendeteksi *docking station* menggunakan persepsi visual yang mampu bekerja pada beberapa tingkat luminasi cahaya tertentu.
3. Mendesain kontroler untuk menggerakan robot menuju *docking station* dan melakukan penyambungan konektor pada *charger* berdasarkan persepsi visual.

Penelitian sistem *auto docking* pada *service robot* menggunakan persepsi visual memiliki beberapa manfaat. Manfaat tersebut adalah untuk membuat robot mampu secara otomatis melakukan pengisian baterai dengan mengenali posisi *docking station* menggunakan masukan data dari sebuah kamera. Sehingga saat posisi *docking* berubah, robot tetap mampu mengenali *docking*. Hasil penerapan sistem ini diharapkan dapat memudahkan pengguna *service robot* dalam melakukan pengisian baterai.

1.4 Batasan Masalah

Sistem yang dibangun dalam penelitian ini dibatasi beberapa hal, antara lain.

1. Area di antara robot dengan *docking station* tidak ada suatu penghalang.
2. *Docking station* berada dalam satu ruangan dengan *service robot*.

Halaman ini sengaja dikosongkan

BAB 2

DASAR TEORI DAN KAJIAN PUSTAKA

Bab ini membahas tentang teori penunjang dan metoda yang dipakai untuk membangun sistem serta membahas mengenai kajian pustaka yang berisi penelitian sebelumnya yang dijadikan sebagai referensi. Teori penunjang yang dibahas mengenai *service robot* RITS-01, pengolahan citra digital, *fuzzy logic controller*, sistem *auto docking* dan strategi *docking* yang dilakukan penelitian sebelumnya.

2.1. *Service Robot* RITS-01

Pada penelitian ini *service robot* yang digunakan adalah robot RITS-01 yang berfungsi untuk melakukan proses pembersihan lantai di gedung pusat robotika ITS [7]. Robot ini juga memiliki fungsi sebagai security robot yang mampu melakukan pengawasan dan pendeteksi orang yang mencurigakan (*interuder*) dengan persepsi visual berbasis kamera. Maka robot dilengkapi dengan kamera untuk sistem navigasi dan juga sistem persepsi suara untuk mengenal perintah suara. Dengan fitur-fitur tersebut maka robot harus menggunakan komputer sebagai kontroler utama dan mikrokontroler sebagai kontroler pendukung.

Gambar 2.1. *Service robot* RITS-01

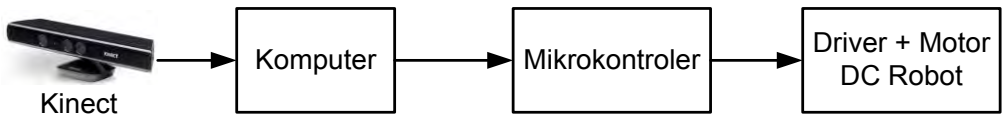
2.1.1. Spesifikasi Robot

Service robot RITS-01 dirancang untuk melakukan pembersihan lantai. Untuk melakukan pembersihan lantai, robot dilengkapi dengan sebuah vaccum cleaner. Aparatur pembersih diletakan pada bagian depan robot dan vaccum cleaner diletakkan di body robot. *Vaccum cleaner* yang digunakan membutuhkan listrik dengan tegangan AC 220 volt sehingga robot dilengkapi dengan DC to AC converter. *DC to AC converter* yang digunakan mampu merubah tegangan DC 12 Volt menjadi tegangan AC 220 Volt.

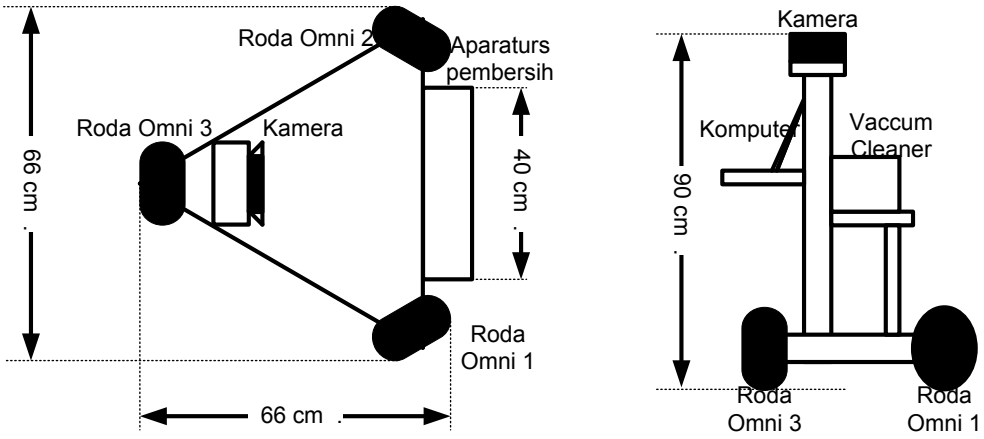
Untuk sumber tenaga dari robot digunakan baterai 12V sebanyak dua buah. Baterai tersebut digunakan untuk mensuplai motor dc, mikrokontroler, kamera, dan beberapa rangkaian pendukung lainya. Penggunaan baterai hingga 24 volt disebabkan motor dc yang digunakan membutuhkan tegangan 24 volt untuk bekerja. Beberapa rangkaian memerlukan tegangan yang lebih rendah sehingga sebelum mensuplai rangkaian tersebut memerlukan sebuah regulator tegangan.

Sistem penggerak yang digunakan adalah brushless motor dc yang telah dilengkapi sebuah driver. Driver ini dikendalikan oleh sebuah mikrokontroler. Namun mikrokontroler hanya mengendalikan driver sesuai perintah kontroler utama. Kontroler utama yang digunakan dalam robot adalah sebuah komputer. Komputer yang digunakan melakukan proses capture gambar, memproses gambar hingga dijadikan masukan sebuah kontroler, menentukan gerakan robot, dan memerintah mikrokontroler untuk menjalankan motor dengan komunikasi serial. Diagram blok hardware dari sistem ditunjukan pada Gambar 2.2.

Roda yang digunakan adalah roda *omnidirectional* yang memiliki diamater 12 cm. Roda *omnidirectional* disusun dengan sudut antar roda sebesar 120° atau juga sering disebut kiwi drive. Secara keseluruhan robot RITS-01 memiliki panjang 66 cm, lebar 66 cm dan tinggi 90 cm. Kamera diletakkan pada bagian atas robot dengan ketinggian 90 cm dari lantai.



Gambar 2.2. Diagram blok service robot RITS-01



Gambar 2.3. Parameter dimensi fisik robot RITS-01

2.1.2. Microsoft Kinect Xbox 360

Robot bekerja dengan persepsi visual berbasis kamera, sehingga sistem navigasi pada robot ini hanya menggunakan kamera sebagai sensornya. Kamera yang digunakan pada robot adalah kamera microsoft kinect xbox 360. Kinect adalah perangkat input untuk mendeteksi gerakan yang diproduksi oleh Microsoft untuk Video Game XBOX 360 dan PC dengan sistem operasi windows. Dengan menggunakan kamera yang mirip dengan webcam memungkinkan untuk mengambil citra berwarna dengan format RGB. Resolusi yang dihasilkan citra RGB sebesat 640x480 pada kecepatan 30fps dan 1280x960 pada kecepatan 12fps.

Depth sensor terdiri dari proyektor laser *infrared* yang dikombinasikan dengan sensor CMOS yang menangkap data dalam bentuk 3D pada kondisi cahaya apapun. Proyektor laser memancarkan cahaya *infrared* dengan pola titik-titik (*dot pattern*) dengan jarak antar titik yang sama. Selanjutnya kamera *infared* mengambil citra yang telah disinari oleh proyektor laser. Jarak antar titik laser yang terdapat pada citra kemudian diproses untuk mendapatkan nilai *depth* nya. Saat obyek dekat, titik laser yang menyinari obyek tersebut lebih banyak dan jaraknya lebih rapat dibandingkan dengan saat obyek jauh. Dengan menggunakan library microsoft kinect SDK, kamera kinect mampu mengukur jarak 80 cm hingga 400 cm pada mode normal. Kamera kinect juga dilengkapi dengan *multi-array microphone* dengan kecepatan sampling 16 KHz.

Gambar 2.4. Kamera kinect untuk x-box 360

Sumber : http://www.microsoft.com/en-us/kinectxbox360/

Gambar 2.5. Bentuk fisik motor AXHM450K-GFH

Sumber : http://catalog.orientalmotor.com/ image?cid=1002&plpver/

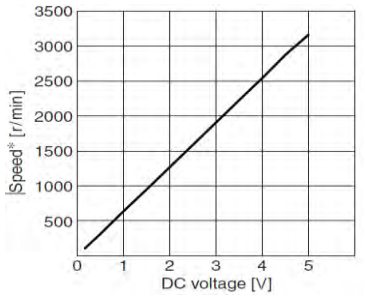
Tabel 2.1. Pergerakan motor terhadap sinyal kontrol

START/STOP	RUN/BRAKE	CW/CCW	Pergerakan Motor
ON (Low level)	ON (Low level)	ON (Low level)	Berputar CW
ON (Low level)	ON (Low level)	OFF (High level)	Berputar CCW
ON (Low level)	OFF (High level)	X	Mengerem
OFF (High level)	ON (Low level)	X	Berhenti

2.1.3. *Brushless Motor DC* AXHM450K-GFH

Sistem penggerak yang digunakan pada robot adalah brushless motor dc AXHM450K-GFH. Motor ini memiliki sebuah gearbox dengan perbandingan poros *output* dengan *input* sebesar 1:30. Motor sudah dilengkapi dengan gearbox sehingga poros output motor bisa langsung disambungkan ke roda.Motor ini menggunakan power supply sebesar 24 volt dengan toleransi 10%.

Motor ini telah dilengkapi dengan sebuah driver. Driver tersebut adalah AXHD50K yang digunakan untuk mengatur kecepatan, menentukan arah putaran motor, melakukan pengereman motor. Sehingga untuk menjalankan fungsi tersebut, driver memiliki empat buah pin input yaitu START/STOP, RUN/BRAKE, CW/CCW dan VRH. VRH merupakan input untuk mengatur kecepatan dan disambungkan dengan pin PWM mikrokontroler. Tabel 2.1. menunjukan pergerakan motor pada tiap kombinasi input.



Gambar 2.6. Grafik kecepatan motor terhadap tegangan drive

Sumber : *Datasheet* AXHD50K

Untuk pengaturan kecepatan, pin VRH diberikan tegangan 0 volt hingga 5 volt. Tegangan 1 volt yang diberikan di pin VRH akan memutar motor dengan kecepatan 650 rpm. Pada setiap kenaikan 1 volt, kecepatan motor naik 650 rpm. Kecepatan tersebut merupakan kecepatan motor sehingga untuk mengetahui kecepatan roda, kecepatan motor tersebut dibagi dengan 30. Grafik kecepatan motor terhadap tegangan drive ditunjukan pada Gambar 2.6.

$$\emptyset_n\ motor = 650\ V_{drive} \tag{2.1}$$

$$V_{drive} = \frac{\emptyset_n\ motor}{650} \tag{2.2}$$

Dimana, Vdrive = tegangan drive (volt)

$\emptyset_n\ motor$ = kecepatan putaran motor yang dinginkan (rpm)

2.1.4. Persamaan Kinematik Model *Kiwi Drive*

Kiwi drive adalah sebuah *holonomic driver* yang menggunakan tiga roda *omnidirectional* dengan konfigurasi berbentuk segitiga. Konfigurasi segitiga ini membuat ukuran *chassis* lebih kecil dari konfigurasi dengan empat roda. Lokasi dan orientasi robot terhadap sebuah *global frame* (x_G, y_G) dapat direpresentasikan dengan koordinat (x_G, y_G, θ) sehingga *global velocity* dapat ditulis sebagai $(\dot{x}, \dot{y}, \dot{\theta})$. *Local frame* berada pada robot itu sendiri dengan pusat koordinat tepat pada *centre of gravity* robot.

Ketiga roda *omnidirectional* pada robot dipasang pada sudut α_i relatif terhadap koordinat local frame (x_L, y_L). Roda ke-1 ditempatkan pada sudut α_1 = 0°, roda ke-2 pada sudut α_2 = 120°, dan roda ke-3 pada sudut α_3= 240° seperti

yang ditunjukan Gambar 2.4. Hubungan *global velocity* dengan kecepatan robot dapat dihitung dengan menggunakan persamaan kinematik pada setiap poros roda. Dalam penelitianya, [7] menggunakan persamaan kinematik model *kiwi drive* seperti yang ditunjukan pada persamaaan (2.3), (2.4), dan (2.5).

$$\dot{\emptyset}_1 = \frac{1}{r}\left(-\sin(\theta + \alpha_1)\cos\theta\,\dot{x} + \cos(\theta + \alpha_1)\cos\theta\,\dot{y} + R\,\dot{\theta}\right) \quad (2.3)$$

$$\emptyset_2 = \frac{1}{r}\left(-\sin(\theta + \alpha_2)\cos\theta\,\dot{x} + \cos(\theta + \alpha_2)\cos\theta\,\dot{y} + R\,\dot{\theta}\right) \quad (2.4)$$

$$\emptyset_3 = \frac{1}{r}\left(-\sin(\theta + \alpha_3)\cos\theta\,\dot{x} + \cos(\theta + \alpha_3)\cos\theta\,\dot{y} + R\,\dot{\theta}\right) \quad (2.5)$$

Dimana , $\emptyset_1$ = kecepatan roda omni 1

$\emptyset_2$ = kecepatan roda omni 2

$\emptyset_3$ = kecepatan roda omni 3

α_1 = penempatan roda omni 1

α_2 = penempatan roda omni 2

α_3 = penempatan roda omni 3

$\dot{x}$ = kecepatan sumbu x_L

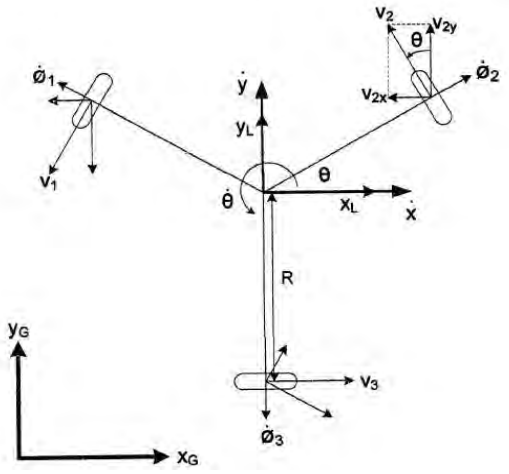
$\dot{y}$ = kecepatan sumbu y_L

θ = orientasi robot terhadap global frame

$\dot{\theta}$ = kecepatan rotasi body robot

R = jarak roda dengan titik tengah robot.

r = jari-jari roda



Gambar 2.6. Diagram kinematik robot *omnidirectional*

Sumber : buku tesis Hendawan Soebhakti [7]

2.2. Pengolahan Citra Digital

Sebuah robot berbasis visual harus memiliki kemampuan mengenali lingkungan sekitarnya dari citra yang diperolehnya dari kamera. Citra tersebut bisa terdapat informasi berupa warna, garis, tulisan, obyek dan lain-lain. Untuk dapat mengetahui informasi tersebut maka diperlukan sebuah pengolahan citra digital. Proses ini dapat menghasilkan informasi yang dicari dari nilai intensitas tiap piksel. Dalam *service robot* berbasis visual, informasi tersebut berkaitan dengan sistem navigasinya. Sehingga untuk bergerak dengan baik informasi yang diperoleh harus sebaik mungkin.

2.2.1. Model Warna *Grayscale*

Citra yang diperoleh dari hasil capture kamera pada umumnya adalah citra berwarna dengan format RGB. Namun saat dilakukan sebuah pengolahan citra biasanya citra RGB tersebut dikonversi menjadi *grayscale*. Perubahan image dari format RGB menjadi format *grayscale* dilakukan dengan menggunakan metode illuminance *grayscale* yang direpresentasikan dalam Persamaan (2.4). Metode ini dilakukan dengan mengalikan nilai intensitas piksel red (R), green (G) dan blue (B) dengan konstanta tertentu dan kemudian dijumlahkan. Hasil penjumlahan tersebut merupakan nilai *grayscale*.

$$I_{gray}(x,y) = 0{,}299I_R(x,y) + 0{,}587I_G(x,y) + 0{,}114I_B(x,y) \tag{2.6}$$

2.2.2. Binerisasi Citra

Dalam beberapa keperluan kita perlu mengkonversi citra yang diperoleh hanya menjadi dua nilai intensitas saja. Proses ini sering disebut binerisasi citra. Proses ini diperlukan sebuah nilai ambang batas atau *threshold* yang nilainya disesuaikan dengan keperluan kita.

$$I_{Bin}(x,y) = \begin{cases} 255, & I_{gray}(x,y) \geq Th \\ 0, & I_{gray}(x,y) < Th \end{cases} \tag{2.7}$$

Th adalah nilai *threshold* yang ditentukan. Dimana proses ini menghasilkan suatu image dengan warna hitam dan putih. Hasil binerisasi ini akan digunakan untuk proses lebih lanjut.

2.3. Local Binary Pattern

Local binary pattern (LBP) memiliki performa yang sangat baik saat digunakan pada beberapa aplikasi seperti klasifikasi tekstur dan inspeksi permukaan. LBP didefinisikan sebagai perbandingan nilai intensitas piksel pusat dengan nilai piksel disekelilingnya (*neighborhood pixel*). Misal pada sebuah citra berukuran 3x3, nilai intensitas pada pusat citra dibandingkan dengan nilai sekelilingnya. Dengan cara mengurangkan nilai piksel pada pusat citra dengan nilai piksel disekelilingnya, jika hasilnya lebih atau sama dengan 0 maka biberi nilai 1 dan jika hasilnya kurang dari 0 maka diberi nilai 0.

Dari hasil konversi LBP tersebut didapatkan nilai biner pada tiap piksel disekeliling piksel pusat. Untuk mengetahui nilai LBP dari piksel pusat tersebut maka nilai biner tiap piksel disekelilingnya dijadikan nilai desimal. Namun MSB dari data biner tersebut belum kita ketahui sehingga terlebih dahulu dicari piksel yang menjadi MSB. Maka untuk menentukan MSB digunakan piksel yang memiliki intensitas tertinggi. Untuk mendapatkan nilai LBP digunakan persamaan sebagai berikut.

$$s(x+i,y+i)=\begin{cases}0,\ I_{gray}(x+i,y+i)<I_{gray}(x,y)\\1,\ I_{gray}(x+i,y+i)\geq I_{gray}(x,y)\end{cases};\ -1\leq i\leq 1 \qquad (2.8)$$

$$max_{neighbor}=\max s(x+i,y+i) \qquad (2.9)$$

$$I_{LBP}(x,y)=\sum_{i=-1}^{1} s(x+i,y+i).2^{mod(i-Max_{neighbor},8)} \qquad (2.10)$$

2.4. *Histogram Matching*

Histogram adalah representasi grafis untuk distribusi warna dari citra digital atau juga merupakan sebuah diagram atau kurva yang menyatakan jumlah kemunculan setiap tingkat keabuan warna. Histogram tersusun dari sumbu vertikal merupakan representasi piksel dengan nilai tonal dari tiap-tiap deret bin

pada sumbu axis horizontalnya. Atau dengan kata lain deret bin pada sumbu horizontal menunjukan nilai intensitas piksel dan sumbu vertikal menunjukan jumlah piksel yang memiliki intensitas tertentu.

Histogram tersebut dapat digunakan untuk mewakili suatu citra tertentu. Sehingga untuk mencari kemiripan suatu gambar dapat dilakukan dengan mencocokan histogramnya. *Histogram matching* adalah salah satu metode dalam pengolahan citra digital yang berfungsi untuk mengetahui kesamaan atau kemiripan antara dua histogram atau lebih. *Histogram matching* pada umumnya dapat dilakukan dengan menggunakan persamaan histogram intersection.

$$H_{match}(H_1, H_2) = \sum_I \frac{\min(H_1(I), H_2(I))}{H_1(I)} \quad (2.11)$$

Dimana, H_{match} = Nilai kesamaan histogram

H_1 = Histogram yang dijadikan acuan atau *template*

H_2 = Histogram yang akan dicari kesamaanya dengan *template*

2.5. *Fuzzy Logic Controller*

Fuzzy logic controller dikategorikan dalam kontrol cerdas (*intelligent control*). Unit logika fuzzy memiliki kemampuan menyelesaikan masalah perilaku sistem yang komplek, yang tidak dimiliki oleh kontroler konvensional. Keuntungan menggunakan *fuzzy logic controller* adalah pendesain tidak memerlukan model matematika dari plan yang akan dikontrol. Yang diperlukan dalam mendesain *fuzzy logic controller* adalah pengetahuan terhadap tingkah laku plan. Sehingga kontroler ini sering digunakan pada sebuah plant yang sulit mendapatkan model matematika. Selain itu desain relatif lebih sederhana dari kontroler cerdas lainya.

Fuzzy logic controller dibuat melalui tiga langkah utama yaitu fuzzifikasi, evaluasi rule atau *inference*, dan defuzzifikasi. Fuzzifikasi merupakan proses untuk mengubah variabel crisp (variabel numerik) menjadi variabel fuzzy (variabel linguistik). Nilai masukan yang dikuantisasi sebelumnya diolah oleh kontroler logika fuzzy dan kemudian diubah ke dalam variabel fuzzy. Melalui membership function (fungsi keanggotaan) yang telah disusun, maka dari

masukan tersebut didapatkan derajat keanggotaan bagi masing-masing masukan. Gambar 2.6 merupakan salah satu contoh penerapan *fuzzy logic controller*.

Membership function adalah suatu fungsi (kurva) yang menunjukkan pemetaan titik-titik input data ke dalam nilai keanggotaannya (derajat keanggotaan) yang memiliki interval antara 0 sampai 1. *Membership function* yang digunakan pada umumnya adalah fungsi segitiga, trapesium, sigmoid, dll. Fungsi segitiga menghasilkan nilai keanggotaan yang linier. Persamaan (2.12) menunjukan cara untuk mendapatkan nilai keanggotaan dengan fungsi segitiga.

$$\mu_{input} = 1 - \frac{2 * \left| x_{crisp} - b \right|}{c - a} \tag{2.12}$$

Dimana, μ_{input} = derajat keanggotaan

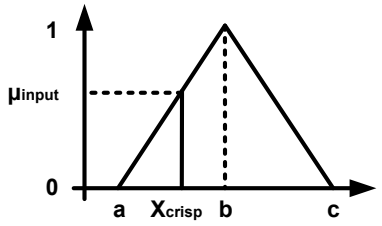
x_{crisp} = nilai crisp *input*

a = nilai awal segitiga

b = nilai tengah segitiga

c = nilai akhir segitiga.

Persamaan (2.12) berdasarkan mengacu segitiga pada Gambar 2.7. Setelah dilakukan proses fuzzifikasi, langkah berikutnya adalah proses evaluasi rule atau *inference*. Masing-masing rule merepresentasikan hubungan antara *input* dan *output* menggunakan aturan *if-then*. Proses evaluasi rule mengevaluasi derajat keanggotaan tiap-tiap fungsi keanggotaan himpunan fuzzy masukan ke dalam basis aturan yang telah ditetapkan. Dengan aturan tersebut kemudian disusun sebuah tabel evaluasi rule untuk mempermudah proses pemrograman kontrol. Tujuan dari evaluasi aturan ini adalah menentukan derajat keanggotaan dari keluaran fuzzy.



Gambar 2.7. Segitiga yang digunakan untuk fuzzifikasi

Setelah mendapatkan derajat keanggotaan dan melakukan evaluasi rule maka langkah berikutnya adalah proses defuzzifikasi. Proses defuzzifikasi adalah proses mengubah variabel fuzzy menjadi variabel *crisp*. Hingga proses evaluasi rule nilai yang didapatkan masih berupa variabel fuzzy (linguistik) sehingga agar perlu dikonversi lagi menjadi variabel crisp (numerik). Defuzzifikasi dapat dilakukan dengan beberapa cara salah satunya adalah *centre of gravity*. Defuzzifikasi dengan *centre of gravity* dilakukan dengan Persamaan (2.13).

$$z_{output} = \frac{\sum \mu c(z) z dz}{\sum \mu c(z) dz} \tag{2.13}$$

2.6. Perkembangan *Service Robot*

Aplikasi robot sudah banyak dikembangkan pada beberapa bidang seperti pada militer, industri, pendidikan, kesehatan dan lain-lain. Menurut [1] aplikasi interaksi manusia dan robot akan semakin berkembang Aplikasi berkembangnya teknologi robot salah satunya adalah aplikasi robot pada rumah tangga seperti *service robot* atau robot pembantu. Robot pembantu atau *service robot* adalah sebuah robot yang mampu melakukan beberapa kegiatan sehari-hari seperti membersihkan ruangan atau mengantarkan benda dari suatu ruangan ke ruangan lain.

Beberapa peneliti mengembangkan aplikasi *service robot* pada berbagai bidang antara lain *service robot* untuk catering [8], untuk pembersih rumah [9], sebagai pelayan restoran [10] dan untuk membersihkan lingkungan laboraturium [7][11]. Masing-masing *service robot* memiliki sistem kerja yang berbeda-beda karena memiliki fungsi utama yang berbeda. Untuk aplikasi pelayan restoran robot mampu untuk menampilkan menu dan mengantar pesanan pelanggan. Pergerakan robot juga dibatasi pada meja pelanggan dengan dapur restoran. Berbeda dengan robot pembersih yang harus menjangkau seluruh ruangan.

2.7. Perkembangan Sistem *Auto Docking*

Robot yang telah memiliki sistem *auto docking* telah dikembangkan oleh beberapa peneliti [12] dan [3]. Pada penelitianya telah menerapkan sistem *auto*

docking pada mobile robot menggunakan sensor ultra sonic. Namun sistem yang menggunakan ultrasonic ini memiliki kelemahan dimana lokasi *docking* harus ditentukan dan posisinya tetap karena tidak memiliki sistem secara visual. Kemudian [4] menggunakan sebuah infrared *rangefinder* untuk sensor robot dalam menemukan *docking*nya. Namun sistem tersebut memiliki kelemahan yang sama dengan sistem yang menggunakan ultrasonic.

Untuk memperbaiki kekurangan tersebut maka sebuah sistem *auto docking* harus dilengkapi dengan sebuah sistem visual. *Mobile robot* yang diteliti oleh [5] menggunakan sebuah kamera untuk melakukan *docking* secara otomatis. Proses *docking* yang dibuat menggunakan sebuah acuan dari dua buah penanda. Penanda yang digunakan dibuat dari lampu halogen yang diletakan pada *docking* dengan susunan vertikal. Prinsip ini mengacu pada prinsip yang digunakan *range light*. *Range light* biasanya digunakan untuk navigasi kapal agar dapat melewati jalur yang aman. Sistem ini memiliki sebuah kelemahan yaitu sulit untuk menentukan jarak antara *docking station* dengan robot. Selain itu, pengaruh gangguan sumber cahaya lain dapat mempengaruhi proses penentuan posisi.

Pada penelitianya, [6] menggunakan *artificial landmark* untuk menandakan lokasi *docking station*. *Artificial landmark* diletakan diatas *docking station* sehingga lokasi *artificial landmark* dapat mewakili lokasi *docking station*. *Artificial landmark* yang digunakan berupa lingkaran berwarna. Pengukuran jarak antara robot dengan landmark ditentukan dengan mengukur diamater landmark ke arah vertikal.

(a) (b)

Gambar 2.8. *Docking station* yang dikembangkan peneliti sebelumnya
(a) Menggunakan *marker* [5] (b) Menggunakan *artificial landmark* [6]

Diameter diperoleh dalam satuan jumlah piksel dan dikonversi hingga mendapat jarak robot hingga *docking station*. Posisi robot terhadap *docking station* bisa diketahui dari diamater *landmark* arah horizontal. Kelemahan sistem ini adalah perubahan luminasi cahaya akan berpengaruh terhadap kemampuan sistem menemukan *landmark*. Jika lingkaran yang dideteksi tidak utuh akibat perubahan luminasi akan berakibat kesalahan penentuan jarak dan posisi *landmark*.

2.8. Penelitian Object Detection

Object detection bisa diterapkan dalam menemukan *docking station* berbasis visual. Beberapa cara deteksi obyek bisa diaplikasikan dalam sistem ini seperti pixel based *template* matching, *speeded up robust features* (SURF), dan color *histogram matching*. Pada makalah [13] disebutkan bahwa pixel based *template* matching memproses nilai piksel sehingga terpengaruh perubahan luminasi. Pengaruh yang sama juga terjadi pada color histogram tetapi color *histogram matching* lebih baik membedakan orientasi obyek dibandingkan dengan *template* matching.

SURF memiliki akurasi tinggi terhadap perbedaan orientasi walaupun masalah luminasi cahaya masih belum teratasi. Pada penelitian [14], pendeteksian dengan SURF diawali dengan cara merubah image RGB menjadi *grayscale*. Selanjutnya menghitung *integral image* dan *hessian matrix* menggunakan fungsi SURF pada opencv untuk mendapatkan *interest point*. Setelah *interest point* ditemukan maka dilakukan perhitungan *feature point*, *nearbor response* dan orientasi *interest point*. Proses diakhiri dengan proses matching dengan KD-tree. Hasil pendeteksian dengan SURF mencapai 5 detik sehingga untuk aplikasi mobile robot kurang tepat untuk direalisasikan.

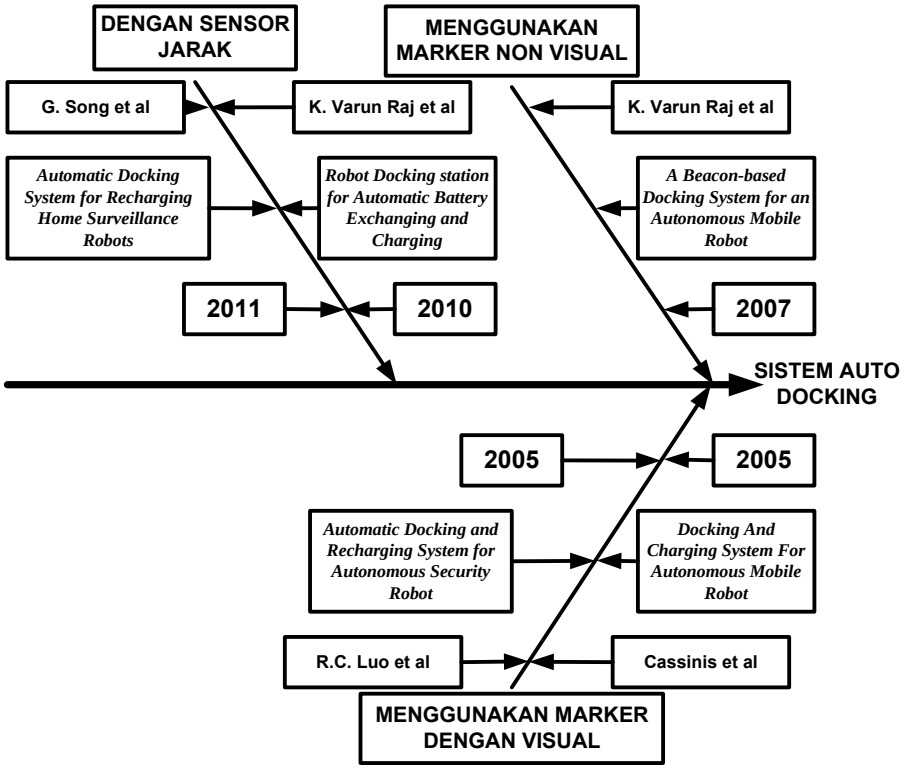
2.9. Perkembangan Penelitian Local Binary Pattern

Kendala perubahan luminasi cahaya dapat diselesaikan dengan LBP. LBP menurut [15] mampu mengetahui pola suatu gambar pada perbedaan rotasi. Selain itu LBP juga tahan terhadap perubahan luminasi cahaya karena menggunakan perbandingan nilai piksel [16]. LBP memiliki performa yang sangat baik saat

digunakan pada beberapa aplikasi seperti klasifikasi tekstur dan inspeksi permukaan.

Penelitian [15] menggunakan LBP untuk tracking obyek. Setelah melakukan konversi *grayscale* menjadi LBP selanjutnya membuat histogram dari citra LBP tersebut. Histogram LBP tersebut digunakan untuk mengekstrak pola-pola yang berada di sebuah citra bergerak. Langkah selanjutnya adalah melakukan perhitungan histogram pola-pola yang telah didapatkan.

Pada penelitian [17], untuk menemukan suatu obyek pada sebuah citra dilakukan *histogram matching*. *Histogram matching* dilakukan dengan melakukan perbandingan antara histogram obyek dan histogram citra sumber. Dalam penelitian tersebut histogram yang dibandingkan terdiri dari dua tahap. Tahap pertama adalah menggunakan histogram citra warna RGB dan dilanjutkan dengan tahap kedua menggunakan citra LBP. Untuk mempercepat proses komputasi pada penelitian menggunakan *histogram bin reduction*.



Gambar 2.9. *Fishbone diagram* penelitian sistem *auto docking*

BAB 3

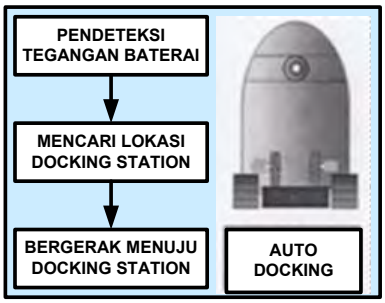
METODA PENELITIAN

Bab ini membahas tentang beberapa tahap pembuatan sistem *auto docking* pada *service robot* menggunakan persepsi visual. Tahap-tahap tersebut meliputi penentuan kondisi dimana sistem *auto docking* mulai dijalankan, proses menemukan *docking station* hingga menjalankan robot untuk menyambungkan konektor. Sistem *auto docking* dengan persepsi visual dibuat sesuai dengan diagram blok yang ditunjukan pada Gambar 3.1. Setelah mendeteksi bahwa proses *auto docking* harus dilakukan, robot akan mencari *docking station*. Sistem deteksi obyek yang dibuat menggunakan local binary pattern (LBP) histogram. Setelah *docking station* ditemukan selanjutnya robot akan bergerak menuju *docking station* untuk menyambungkan konektor.

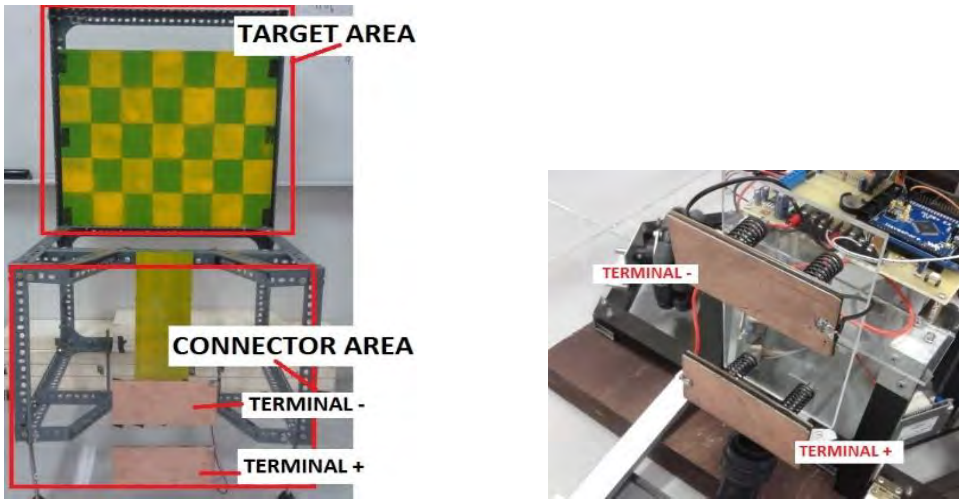
3.1. Desain *Docking Station*

Robot RITS-01 belum memiliki *docking station* sehingga terlebih dahulu kita harus membuat *docking station* terlebih dahulu. *Docking station* yang dibuat terlebih dahulu disesuaikan dengan dimensi robot. *Docking station* memiliki terminal yang digunakan mengalirkan listrik ke robot. Terminal ini terletak di bagian bawah *docking station* karena disesuaikan dengan dimensi robot. Namun kamera yang digunakan robot berada di ketinggian 90 cm sehingga saat posisi dekat kamera tidak mampu melihat posisi konektor.

Permasalahan ini diselesaikan dengan membagi *docking station* menjadi dua bagian yaitu bagian terminal dan bagian yang dilihat kamera (target area). Ketinggian target area diseuaikan dengan ketinggian kamera robot. Dengan ketinggian target area yang sama dengan kamera diharapkan saat mendekat ke *docking station*, kamera masih mampu melihat *docking station*.



Gambar 3.1. Diagram blok sistem *auto docking* menggunakan persepsi visual



(a) (b)

Gambar 3.2. Hardware pendukung sistem auto docking,
(a). Desain *docking station* (b). Konektor pada robot

Pada bagian connector area terdapat sebuah *power supply* yang akan digunakan untuk mengisi baterai. Tegangan yang dihasilkan *power supply* tersebut adalah 26,4 volt. Output *power supply* terhubung ke sebuah terminal yang berada di depan connector area. Dengan posisi tersebut maka akan mempermudah penyambungan konektor oleh robot. Sebab konektor yang berada pada robot juga berada di depan. Sehingga proses penyambungan dapat dilakukan dengan menabrakan robot ke *docking station*.

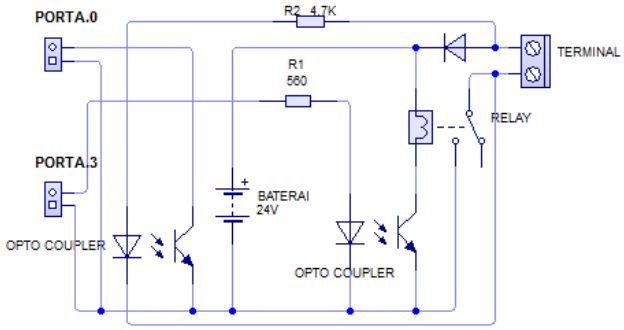
3.2. Sistem Pembaca Tegangan Baterai

Untuk mengetahui kapan sistem *auto docking* harus dijalankan, maka dibuat sistem pembaca tegangan baterai. Tegangan baterai bisa digunakan untuk menentukan kapasitas baterai yang tersisa dalam baterai. Perbandingan tegangan dengan kapasitas tertera dalam *datasheet* tiap-tiap baterai. Sehingga dalam sistem pembaca tegangan baterai ini akan mengacu pada data tersebut.

Sistem pendeteksi baterai pada penelitian ini dibuat menggunakan analog to digital converter. Saat tegangan yang terbaca sudah mencapai tegangan minimal untuk bekerja maka robot akan melakukan proses *auto docking*. Saat baterai sudah terisi penuh maka proses pengisian baterai akan dihentikan dan robot siap kembali bekerja. Sistem pembaca tegangan baterai dirancang berdasarkan diagram blok pada Gambar 3.3. Sistem membaca dua buah data yaitu besar tegangan dan kondisi *charge*/*discharge*. Data selanjutnya diproses oleh sistem pengambilan keputusan. Hasil dari proses tersebut adalah pengiriman sinyal ke kontroler robot untuk melakukan *auto docking* atau bekerja.

Dalam desain sistem *auto docking* ini tegangan minimal untuk robot bekerja adalah 23,5 volt. Saat sistem mendeteksi tegangan baterai sudah sama atau dibawah 23,6 volt saat posisi *discharge* maka robot akan menjalankan proses *auto docking*. Tegangan minimal tersebut dipilih berdasarkan dua parameter yaitu batas minimal motor dc robot dapat bekerja dan tegangan *discharge* baterai saat menyisakan durasi kurang dari 20 sebelum mencapai tegangan *cut off*. Tegangan *discharge* tersebut dapat dilihat pada *datasheet* baterai yang digunakan.

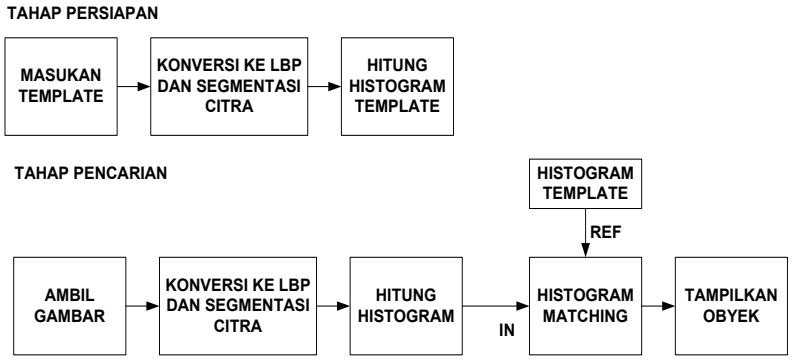
Gambar 3.3. Diagram blok sistem pembaca tegangan baterai



Gambar 3.4. Rangkaian konektor terminal *charger* dan baterai

Motor dc yang digunakan robot menurut *datasheet* mampu bekerja hingga tegangan 21,6 volt. Namun pada saat melakukan tes motor, motor sudah tidak mampu berputar dengan maksimal pada tegangan 23,5 volt. Sehingga tegangan minimal motor dc yang digunakan adalah 23,5 volt. Baterai yang digunakan adalah panasonic LC-R127R2 yang memiliki tegangan *cut off* sebesar 10,8 volt. Menurut *datasheet* [18], saat tegangan *discharge* bernilai 11,4 volt sisa waktu sebelum mencapai tegangan *cut off* adalah 20 menit. Jumlah baterai yang digunakan sebanyak 2 buah sehingga tegangan *discharge* yang digunakan adalah 22,8 volt. Dari kedua tegangan tersebut yang digunakan sebagai tegangan minimal robot adalah yang nilainya lebih besar yaitu 23,5 volt. Tegangan maksimal adalah tegangan baterai saat terisi penuh pada proses pengisian. Menurut *datasheet* baterai, tegangan maksimal yang digunakan adalah 13,2 volt atau jika menggunakan dua baterai maka tegangan maksimal adalah 26,4 volt. Saat melakukan pengisian dan tegangan baterai sudah melewati tegangan maksimal maka proses pengisian harus dihentikan.

Untuk mengetahui kondisi *discharge* atau *charge*, pada rangkaian konektor terdapat sebuah optocoupler yang mendeteksi adanya arus listrik pada terminal *charger*. Saat ada arus mengalir, optocoupler aktif sehingga PORTA.0 mikrokontroler terhubung ke terminal negatif baterai atau berada pada kondisi logika 0. Jadi saat PORTA.0 bernilai logika 0, robot berada pada kondisi charging dan berlaku sebaliknya. Saat terdeteksi kondisi charging, PORTA.3 mengaktifkan relay untuk menyambungkan *charger* dengan baterai. PORTA.3 juga berfungsi untuk memutus sambungan sementara saat sistem membaca tegangan baterai.



Gambar 3.5. Diagram blok sistem pendeteksi *docking station*

3.3. Sistem Pendeteksi *Docking Station*

Dalam melakukan proses pendeteksian *docking station*, sistem dibagi menjadi dua tahap. Tahap pertama adalah tahap persiapan yang berfungsi untuk mempersiapkan *template*. tahap kedua adalah tahap pencarian yang berfungsi untuk mencari lokasi *docking station* dari *template* yang sudah didapatkan.

3.3.1. Tahap Persiapan

Sebelum melakukan pencarian *docking station*, robot harus mengenali bentuk *docking station* terlebih dahulu. Proses ini dilakukan dengan menyapkan gambar *template* dari *docking station*. *Template* yang digunakan adalah bagian target area dari *docking station*. Jadi pada tahap ini pertama-tama disiapkan gambar target area dari *docking station* yang digunakan. *Template* yang digunakan ditunjukan pada Gambar 3.6.

Citra *template* yang masih dengan format RGB kemudian dikonversi menjadi *grayscale*. Setelah citra *grayscale* didapatkan kemudian dilakukan proses konversi dari *grayscale* menjadi LBP. Proses ini dilakukan menggunakan Persamaan (2.8), (2.9), dan (2.10) dimana hasil akhir dari proses ini adalah didapatkanya citra LBP dari *template*. Citra LBP memiliki resolusi yang sama dengan citra *template* yang digunakan. Untuk meningkatkan akurasi maka citra LBP yang diperoleh dibagi menjadi beberapa bagian. Masing-masing bagian yang dibentuk akan digunakan untuk pencocokan dengan citra hasil capture kamera. Semakin banyak area yang dibentuk maka ketelitian akan semakin tinggi tetapi

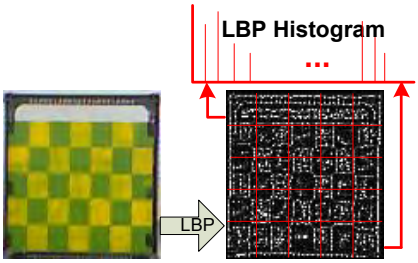
memakan waktu komputasi lebih lama. Proses penentuan luas area segmentasi dilakukan dengan menggunakan Persamaan (3.1) dan (3.2).

$$segmen_{width} = \frac{template.width}{n} \quad (3.1)$$

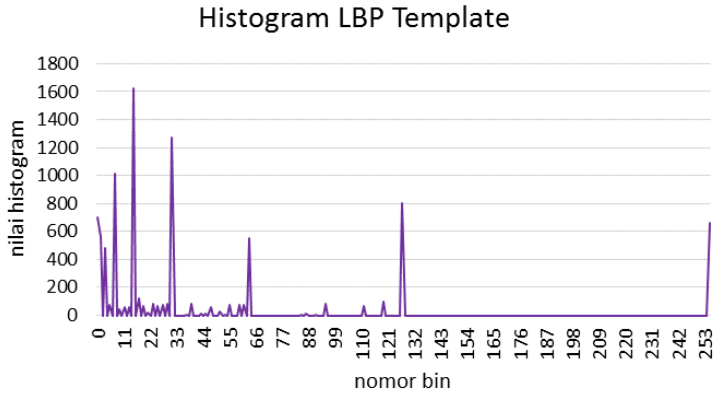
$$segmen_{heigth} = \frac{template.heigth}{n} \quad (3.2)$$

Dimana, $segmen_{width}$ = lebar area (piksel)
$segmen_{width}$ = tinggi area (piksel)
n = jumlah segmentasi

Gambar 3.6. *Template* yang digunakan



Gambar 3.7. Segmentasi citra LBP *template*



Gambar 3.8. Histogram LBP citra *template*

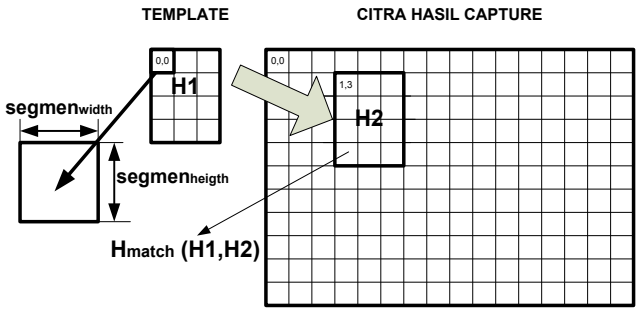
Masing-masing area citra LBP berisikan pola-pola LBP yang berada dalam *template*. Untuk mendapatkan jumlah dari masing-masing pola yang muncul adalah dengan menghitung histogram dari citra LBP. Setelah histogram LBP didapatkan kemudian dilakukan proses sorting dari nilai dominan hingga paling kecil. Sehingga setelah proses *sorting* selesai akan didapatkan histogram yang memiliki data urut dari nilai terbesar hingga terkecil.

3.3.2. Tahap Pencarian

Dalam kondisi normal *docking station* diletakkan di lantai dengan posisi berdiri. Kondisi lantai dalam ruang pengujian cenderung datar dan tidak ada tanjakan atau turunan. Dalam posisi kamera normal menghadap ke depan maka *docking station* pasti selalu berada di tengah-tengan layar pada sumbu vertikal. Sehingga untuk menghemat waktu pencarian, area yang di proses berada pada sekitar sumbu horizon. Dengan cara ini lokasi *docking station* pada sumbu vertikal bisa dianggap tetap dan sistem hanya mencari lokasi *docking station* pada sumbu horizontal.

Pada tahap pencarian citra yang didapatkan dari kamera diproses dengan cara yang sama dengan tahap pertama hingga mendapatkan citra LBP. Dari proses *grayscale* hingga persamaan LBP cara yang digunakan tidak ada perbedaan. Perbedaan baru terjadi pada proses segmentasi. Setelah luas area segmentasi ditentukan pada tahap pertama, pada tahap kedua luas area tersebut digunakan untuk menemukan jumlah segmen. Setelah citra terbagi menjadi beberapa bagian maka dilanjutkan dengan prosses perhitungan histogram.

Proses perhitungan histogram yang dilakukan berbeda dengan tahap pertama. Jika di tahap pertama perhitungan histogram dilakukan untuk mencari nilai dominan. Maka pada tahap ini histogram yang dicari hanya yang memiliki nomor bin tertentu. Nomor bin yang digunakan sesuai dengan histogram LBP pada citra *template* yang bernilai maksimal/dominan hingga sejumlah bin yang akan diproses. Jika jumlah bin yang dipakai adalah n buah maka nomor bin yang digunakan adalah n bin dominan. Jumlah bin yang digunakan disesuaikan dengan hasil pengujian yang akan dilakukan. Jumlah bin yang optimal adalah jumlah bin dengan akurasi hasil terbaik dengan kecepatan komputasi tercepat.



Gambar 3.9. Proses *scanning* pada citra kamera

Proses selanjutnya adalah proses pencarian kandidat lokasi *docking station*. Pencarian diawali dengan melakukan *scanning* pada area yang telah ditentukan. Proses *scan* dilakukan dengan sebuah *kernel* yang berukuran sama dengan citra *template* dan dengan hasil segmentasi yang sama. *Scanning* dilakukan pada tiap koorninat dengan interval sebesar luas area hasil segmentasi.

Setiap *kernel* pada tiap-tiap koordinat dilakukan proses *histogram matching*. *Histogram matching* yang dilakukan menggunakan Persamaan (2.11). Hasil proses *histogram matching* kemudian dipilih pada nilai yang melewati *threshold*. Jika ada banyak area yang melewati *threshold* maka hanya akan dipilih beberapa area saja. Masing-masing area yang terpilih selanjutnya dijadikan kandidat lokasi *docking station*. Setelah mendapatkan area yang dijadikan kandidat lokasi *docking station* selanjutnya adalah melakukan pencarian kasar.

Pencarian kasar berfungsi untuk mempersempit area yang menjadi kandidat lokasi *docking station*. Pada area dimana *docking station* berada seharusnya terdapat banyak kandidat yang saling berhimpit. Dari fenomena ini pencarian kasar dilakukan dengan mencari area kandidat yang berdekatan dengan kandidat lain sebanyak n buah. Nilai n ini berpengaruh terhadap ketelitian proses deteksi. Jika terdapat area yang memiliki kandidat saling berdekatan dengan n buah kandidat maka area tersebut dinyatakan sebagai lokasi *docking station*.

Jika pencarian kasar berhasil menemukan *docking station* maka proses pencarian dilakukan dengan proses pencarian halus. Lokasi *docking station* dapat ditemukan walaupun hanya dengan pencarian kasar. Namun selain lokasi *docking station*, orientasi *docking station* juga dicari dalam proses ini. Jadi pencarian halus berfungsi untuk menentukan orientasi *docking station*. Orientasi *docking station*

didapatkan dari hasil perbandingan panjang sisi kanan dan kiri *docking station*. Orientasi sendiri adalah sudut yang dibentuk antara arah hadap robot saat ditarik garis lurus dengan arah hadap *docking station*. Untuk mempermudah pemahaman orientasi tersebut dapat dilihat pada Gambar 3.10.

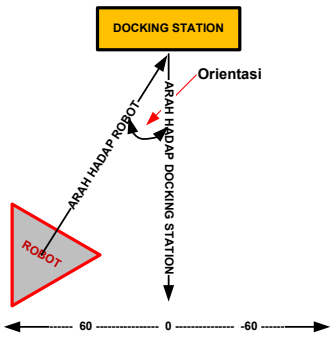
Pencarian halus dilakukan dengan menggunakan format warna HSV. Namun hanya nilai hue nya saja yang digunakan. Pada area hasil pencarian kasar dicari nilai hue paling dominan. Kemudian dari titik tengah area tersebut dilakukan *scanning* ke arah horizontal untuk mengetahui lebar *docking station*. Setelah lebar diketahui maka dilakukan *scan*ing ke arah vertikal pada kedua sisi *docking station*. Hasil *scanning* ini yang akan digunakan untuk mengetahui panjang masing-masing sisi *docking station*. Proses pencarian halus diilustrasikan oleh Gambar 3.11.

3.3.3. Penentuan Masukan Kontroler

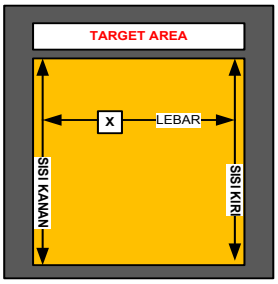
Untuk menjalankan robot maka beberapa parameter hasil pendeteksian *docking station* dimasukan ke dalam sebuah kontroler. Masukan yang dibutuhkan robot adalah lokasi sumbu horizontal *docking station*, orientasi *docking station*, dan jarak robot dengan *docking station*. Tiap-tiap masukan memberikan pengaruh yang berbeda terhadap robot. Sehingga untuk pergerakan tertentu tidak semua masukan tersebut digunakan.

Masukan yang pertama adalah lokasi koordinat sumbu x (horizontal) *docking station*. Dalam penulisan buku ini untuk menunjukan lokasi koordinat sumbu x akan digunakan istilah nilai X. Nilai X merupakan titik tengah area yang didapatkan dari hasil pencarian kasar. Dengan resolusi citra dari kamera sebesar 640 x 480 piksel maka nilai X akan berkisar dari 0 hingga 639.

Masukan kedua adalah orientasi atau arah hadap *docking station* terhadap robot. Nilai orientasi didapatkan dari hasil perhitungan panjang *docking station* pada tiap sisi dalam satuan piksel. Dalam penulisan buku ini untuk menunjukan nilai orientasi akan digunakan istilah nilai O. Untuk menentukan nilai O dari panjang tiap-tiap sisi dilakukan pengujian awal.



Gambar 3.10. Penentuan orientasi

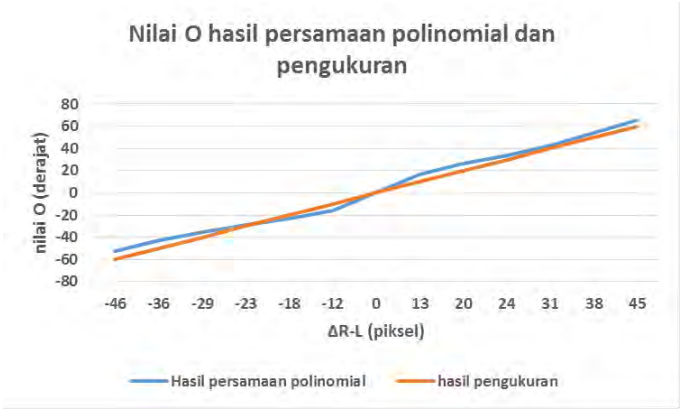


Gambar 3.11. Ilustrasi pencarian halus

Tabel 3.1. Nilai orientasi dari selisih sisi kanan dan kiri *docking station*

Orientasi	**Selisih (sisi kanan-Kiri)**	**Orientasi**	**Selisih (sisi kanan-Kiri)**
-60	-46	10	13
-50	-36	20	20
-40	-29	30	24
-30	-23	40	31
-20	-18	50	38
-10	-12	60	45

Dari data pada Tabel 3.1 selanjutnya dibuat sebuah persamaan polinomial untuk mendapatkan sudut orientasi dari selisih pengukuran panjang sisi kanan dan sisi kiri *docking station*. Persamaan yang didapatkan ditunjukan pada Persamaan (3.3). dari persamaan tersebut kemudian di plot pada sebuah grafik yang ditunjukan pada Gambar 3.12. Nilai negatif menunjukan orientasi arah kiri dan nilai positif menunjukan orientasi arah kanan.



Gambar 3.12. Hubungan antara orientasi dengan selisih kedua sisi *docking station*

$$O = \left(0.0004 * {\Delta_{R-L}}^2\right) + (1.2933 * \Delta_{R-L}) + 1.1013 \quad (3.3)$$

Dimana, O : sudut orientasi (°)

Δ_{R-L} : selisih panjang sisi kanan-sisi kiri (piksel)

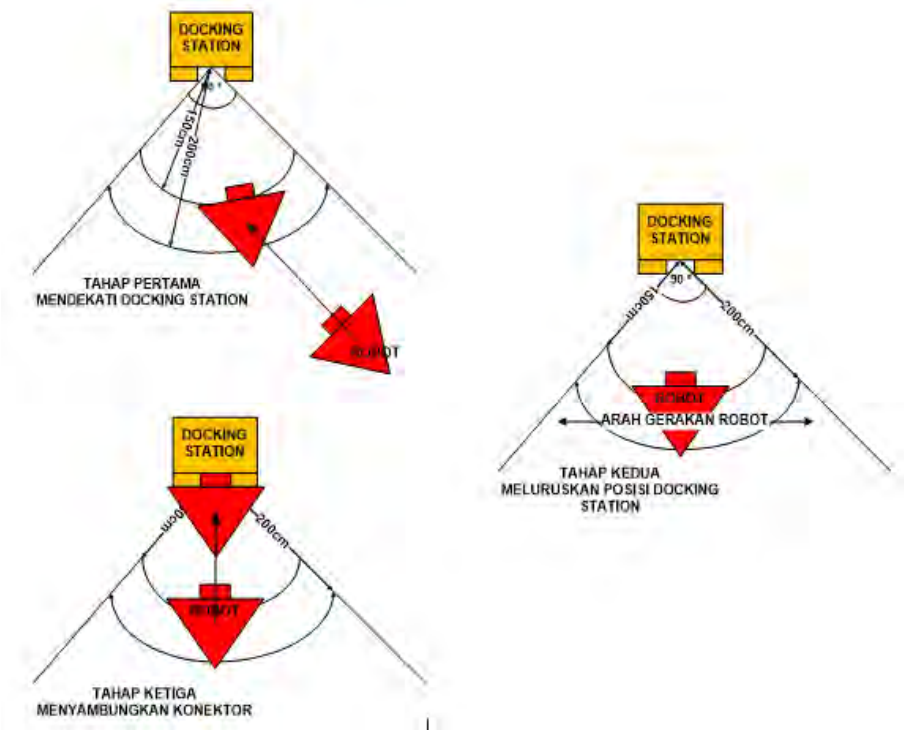
Masukan ketiga adalah jarak antara robot dengan *docking station*. Dalam penulisan buku ini untuk menunjukan jarak antara robot dengan *docking station* akan digunakan istilah nilai D. pengukuran jarak dilakukan dengan citra *depth* dari kamera microsoft kinect. Bagian *docking station* yang diukur adalah bagian koordinat nilai X. Dengan kata lain nilai D adalah jarak antara kamera dengan titik nilai X *docking station*.

3.4. Strategi *Auto Docking*

Penyambungan konektor bisa dilakukan saat posisi *docking station* dan robot saling berhadapan. Namun dalam bergerak menuju *docking station*, robot tidak selalu mampu dalam posisi tersebut. Ada kemungkinan robot berada pada sisi kanan atau kiri *docking station*. Untuk mengatasi permasalahan itu proses *auto docking* dibagi menjadi 3 tahap. Tahap pertama adalah mendekat ke *docking station*, tahap kedua adalah meluruskan posisi dengan *docking station* dan tahap ketiga adalah menyambungkan konektor.

Tahap pertama bertujuan agar robot bergerak mendekati *docking station* hingga jarak 1.5-2 meter. Untuk melakukan proses *docking* hingga penyambungan konektor dalam satu tahap akan mengalami kesulitan saat robot begerak dari arah samping. Tahap kedua proses *auto docking* bertujuan untuk menggerakkan robot agar berada pada posisi saling berhadapan dengan *docking station*. Proses ini dilakukan penyesuaian orientasi obyek agar bernilai 0°. Selain itu nilai X harus berada di tengah atau sama dengan *centre pixel*. Kedua syarat tersebut dipenuhi maka tahap kedua diakatakan selesai dan dilanjutkan dengan tahap tiga.

Tahap ketiga ini dilakukan dengan cara menjalankan robot maju lurus ke depan. Robot akan berhenti saat telah mendeteksi penyambungan konektor. Tahap ketiga dilakukan tanpa adanya panduan arah pergerakan. Hal ini disebabkan jarak antara robot dan *docking station* terlalu dekat sehingga *depth* sensor pada kamera kinect tidak mampu membaca jarak.



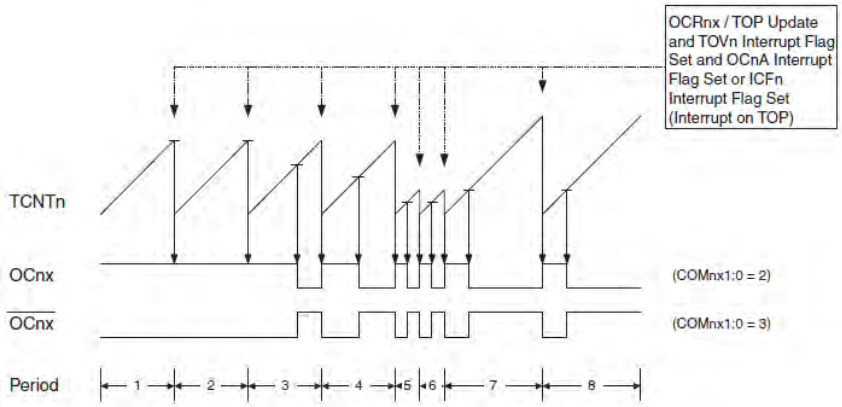
Gambar 3.13. Strategi *auto docking* yang dilakukan

3.5. Pengaturan Kecepatan Robot

Motor DC merupakan penggerak utama dari robot sehingga pengaturan kecepatan robot dilakukan dengan melakukan pengaturan kecepatan motor DC. Rangkaian *driver* yang digunakan motor memerlukan tegangan eksternal untuk mengatur kecepatan motor. Tegangan yang diberikan harus berkisar dari 0 hingga 5 volt dengan arus minimal 1mA [19]. Untuk menghasilkan tegangan tersebut digunakan *pulse width modulation* (PWM) dari mikrokontroler yang digunakan.

Mikrokontroler yang digunakan adalah ATMega128 yang memiliki 8 channel PWM. Tiap channel mampu menghasilkan sinyal PWM dengan duty cycle 0 hingga 100%. Pada mikrokontroler ATMega128, pengaturan duty cycle dilakukan dengan mengatur nilai register OCRnx. Pada penelitian ini digunakan mode 8-bit sehingga nilai register OCRnx bernilai dari 0 hingga 255. Sinyal pwm yang dihasilkan, dikeluarkan melalui port OCnx.

Terdapat dua mode output pada mikrokontroler ATMega128 yaitu non-inverting dan inverting. Pada mode non-inverting nilai Ocnx bernilai 5 volt saat nilai register OCRnx lebih besar dari nilai register TCNTn. Sedangkan pada mode inverting nilai OCnx bernilai 5 volt saat nilai register OCRnx lebih kecil dari nilai register TCNTn. Ilustrasi mode non-inverting dan inverting dapat dilihat pada Gambar 3.14. Tegangan output yang dihasilkan port OCnx pada mode non-inverting didapatkan dari Persamaan (3.4). Sedangkan untuk mode inverting, tegangan output didapatkan dari Persamaan (3.6).



Gambar 3.14. Timing diagram pwm pada mikrokontroler ATMega128

Sumber : *Datasheet* ATMega128

$$Vout_{non\ inv} = \frac{OCRnx * 5}{255} \quad (3.4)$$

$$OCRnx_{non\ inv} = \frac{V_{drive} * 255}{5} \quad (3.5)$$

$$Vout_{inv} = \frac{(255 - OCRnx) * 5}{255} \quad (3.6)$$

$$OCRnx_{inv} = \frac{(5 - V_{drive}) * 255}{5} \quad (3.7)$$

Tegangan tersebut selanjutnya digunakan untuk mengatur kecepatan motor. Perubahan kecepatan motor terhadap tegangan drive dapat dilihat pada grafik yang ditunjukan Gambar 2.6. Kecepatan motor yang didapatkan tidak sama dengan kecepatan roda karena adanya gearbox yang menghubungkan antara motor dan roda. Perdandingan putaran motor dan roda adalah 30:1 sehingga roda berputar sekali jika motor berputar 30 kali. Untuk mendapatkan nilai tegangan drive dari kecepatan motor yang diinginkan digunakan Persamaan (2.1).

Pengaturan kecepatan robot dalam penelitian ini diatur dengan satuan kecepatan m/s. Namun untuk menggerakan motor, kita memerlukan nilai tegangan drive yang diatur dari nilai register OCRnx ATMega128. Sehingga perlu dilakukan proses konversi dari kecepatan global robot menjadi sebuah nilai register. Untuk melakukan konversi tersebut harus diketahui terlebih dahulu persamaan kinematik robot, jarak antar roda robot, diameter roda robot, dan grafik kecepatan motor terhadap tegangan drive.

Dengan menggunakan persamaan kinematik model kiwidrive yang ditunjukan pada Persamaan (2.3), (2.4), dan (2.5) akan didapatkan kecepatan masing-masing roda. Roda yang digunakan pada robot RITS-01 memiliki diameter 12 cm. Sehingga dengan mengetahui diameter roda yang digunakan, akan didapatkan kecepatan roda tersebut dalam satuan RPM. Karena melewati sebuah gearbox, maka kecepatan motor 30 kali lebih cepat dari putaran roda.

$$\emptyset_n motor = \frac{V_n * 1800}{2\,\pi\,r} \quad (3.8)$$

Dimana, $\emptyset_n motor$ = kecepatan putaran motor

V_n = kecepatan global tiap roda

r = jari-jari roda

Untuk mencari nilai tegangan drive, $\emptyset_n motor$ yang telah didapatkan dimasukan ke Persamaan (3.8). Namun hasil perhitungan tersebut masih berupa nilai tegangan. Untuk mendapatkan nilai register OCRnx nilai tegangan drive yang didapat dimasukan ke Persamaan (3.5). Hasil perhitungan tersebut merupakan nilai register OCRnx yang digunakan untuk membangkitkan sinyal pwm.

3.6. Desain Kontroler Pergerakan Robot

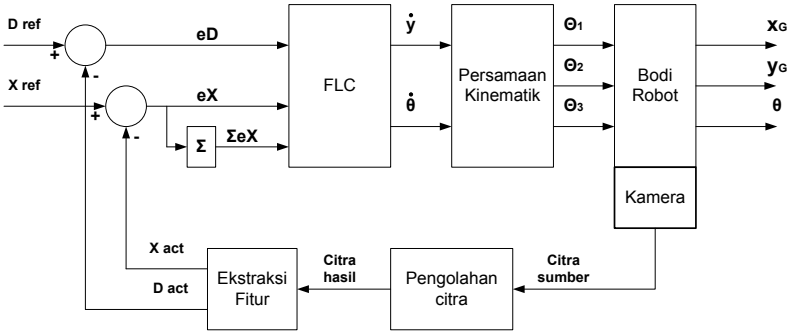
Service robot RITS-01 menggunakan konfigurasi kiwidrive sehingga untuk menggerakan robot diperlukan beberapa masukan. Masukan yang diperlukan antara lain kecepatan sumbu x ($\dot{x}$), kecepatan sumbu y ($\dot{y}$), dan kecepatan rotasi robot ($\dot{\theta}$). Dengan menggunakan Persamaan (2.1),(2.2), dan (2.3) maka akan dihasilkan kecepatan dan arah putaran masing-masing roda.

Untuk mendapatkan $\dot{x}, \dot{y}$, dan $\dot{\theta}$ dari nilai X, D, dan O yang didapatkan dari hasil pengolahan citra maka diperlukan sebuah kontroler. Kontroler yang digunakan dalam penelitian adalah *fuzzy logic controller*. Proses fuzzifikasi dan defuzzifikasi dilakukan dengan persamaan yang sama pada setiap tahap. Namun pada masing-masing tahap menggunakan masukan dan rule yang berbeda. Hal itu didasari pada kebutuhan gerakan di setiap tahap *auto docking*.

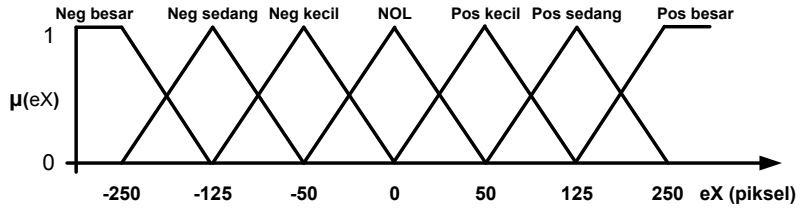
Model matematika dari robot RITS-01 belum diketahui sehingga kontroler yang digunakan adalah *fuzzy logic controller*. Sebab *fuzzy logic controller* tidak memerlukan model matematika dari robot yang akan dikontrol. Dalam mendesain *fuzzy logic controller* yang diperlukan adalah pengetahuan terhadap tingkah laku robot. Beberapa tingkah laku dari robot yang diamati adalah pengaruh nilai X, dan O terhadap arah gerakan robot, dan nilai D terhadap kecepatan robot.

Pada tahap pertama, robot hanya berjalan menuju *docking station* tanpa memperhitungkan orientasi. Pada tahap ini robot hanya menggunakan nilai X sebagai acuan pergerakan robot. Sedangkan nilai D digunakan untuk menentukan pemberhentian robot. Sehingga pada tahap ini masukan kontroler hanya nilai error X (eX) dan error D (eD). Selain masukan error X dan error D, kontroler juga diberikan masukan tambahan yaitu integral error X (ΣeX). Diagram blok kontroler

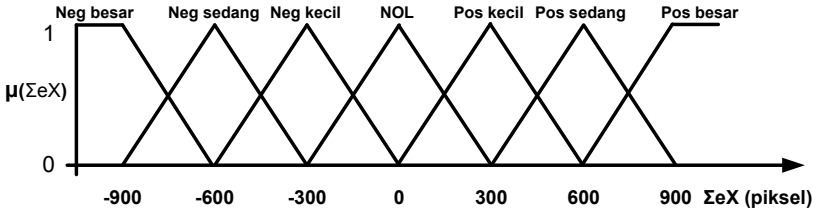
pada tahap pertama ini ditunjukan pada Gambar 3.15. Set poin nilai X yang digunakan adalah 320 karena merupakan titik tengah gambar. Untuk nilai D set poin yang digunakan adalah 175 karena area pemberhentian robot yang diinginkan adalah pada jarak 150 cm hingga 200 cm dari *docking station*.



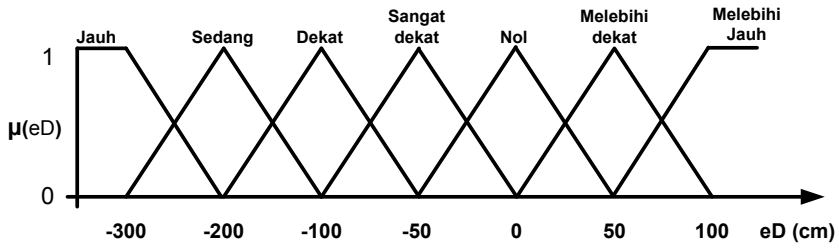
Gambar 3.15. Diagram blok kontroler proses *auto docking* tahap pertama



(a)



(b)



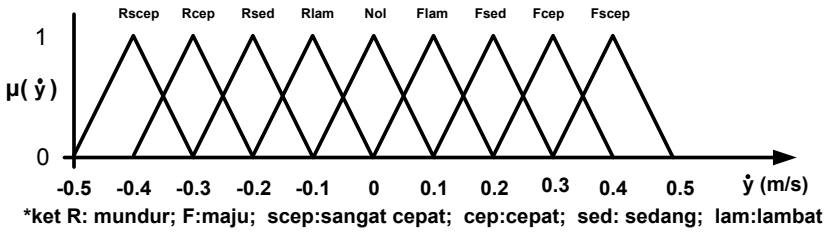
(c)

Gambar 3.16. Input membership function pada kontroler tahap pertama

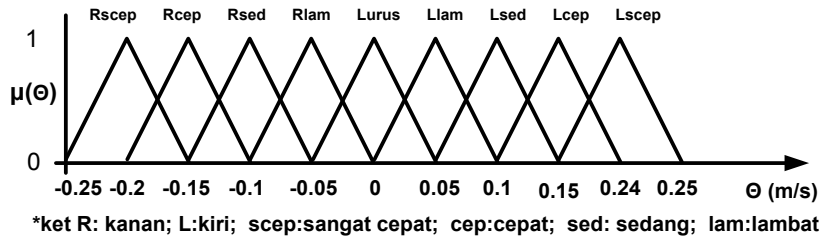
(a). Nilai error X (b). Nilai Σ error X (c). Nilai error D

Proses fuzzifikasi dalam *fuzzy logic controller* yang diterapkan menggunakan *input membership function* seperti yang ditunjukan pada Gambar 3.16. Sedangkan *output membership function* yang digunakan ditunjukan pada Gambar 3.17. Dari *membership function* tersebut akan didapatkan nilai atau derajat keanggotaan. Masing-masing nilai keanggotaan tersebut kemudian dimasukan sebuah proses evaluasi rule. Dalam proses tahap pertama ini, rule yang digunakan disajikan dalam lampiran 1. Dari rule atau aturan yang telah ditetapkan selanjutnya dilakukan proses defuzzifikasi. Proses defuzzifikasi yang dilakukan mengacu pada Persamaan (2.11).

Pada tahap kedua, robot melakukan pergerakan untuk saling berhadapan dengan *docking station*. Nilai D tidak lagi digunakan dalam proses ini karena posisi robot sudah berada di area pemberhentian. Sehingga masukan kontroler yang digunakan hanya nilai X dan O. Namun proses tahap kedua juga dibagi menjadi 2 bagian yaitu meluruskan posisi dan meluruskan arah hadap. Kedua proses dijalankan saling bergantian hingga didapatkan posisi yang lurus dan arah hadap yang tepat. Fenomena posisi dan arah hadap yang tidak tepat ditunjukan pada Gambar 3.18.



(a)



(b)

Gambar 3.17. Output membership function pada kontroler tahap pertama
(a). Output $\dot{y}$ (b). Output $\dot{\theta}$

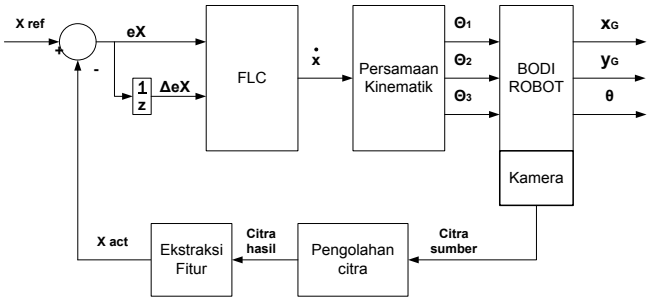
(a) (b)

Gambar 3.18. Fenomena posisi dan arah hadap

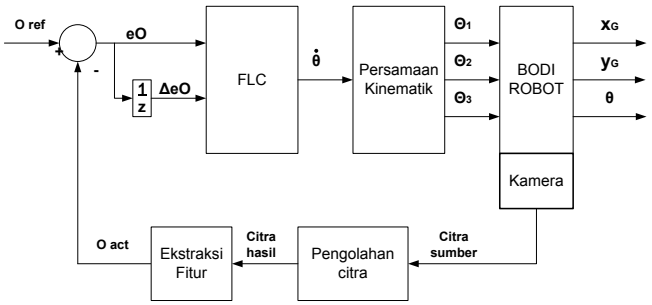
(a). Arah hadap lurus posisi tidak lurus (b). Arah hadap tidak lurus posisi lurus

Dari ilustrasi pada Gambar 3.18 dapat dilihat bahwa arah hadap ditentukan dengan nilai O dan posisi ditentukan oleh nilai X. Sehingga dalam penerapan kontroler tahap kedua ini dibagi menjadi dua bagian dengan kontroler yang berbeda. Untuk meluruskan arah hadap masukan yang digunakan adalah nilai O sedangkan untuk meluruskan posisi digunakan masukan nilai X. Set poin nilai O dan X masing masing adalah 0 dan 320. Proses *auto docking* tahap kedua ini menggunakan kontroler yang terpisah pada tiap tahap. Gambar 3.19 (a) menunjukan diagram blok kontroler untuk meluruskan posisi sedangkan Gambar 3.19 (b) menunjukan diagram blok kontroler untuk meluruskan arah hadap.

Dalam proses *auto docking* tahap kedua terdapat dua bagian sehingga FLC yang digunakan juga ada dua bagian. Bagian pertama yaitu FLC untuk pergerakan meluruskan posisi yang selanjutnya disebut sebagai FLC#1 dan bagian kedua yaitu FLC untuk pergerakan meluruskan orientasi yang selanjutnya disebut sebagai FLC#2. Masing-masing FLC#1 dan FLC#2 memiliki masukan, fungsi keanggotaan, rule, dan output yang berbeda. Namun masih tetap menggunakan dasar persamaan yang sama untuk proses fuzzifikasi dan defuzzifikasi.



(a)

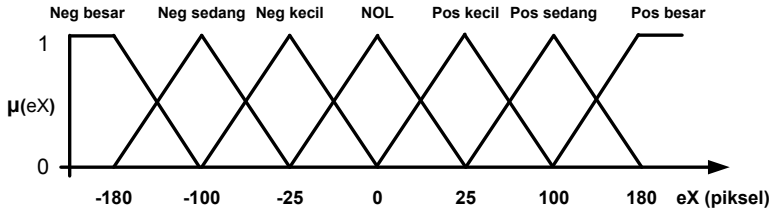


(b)

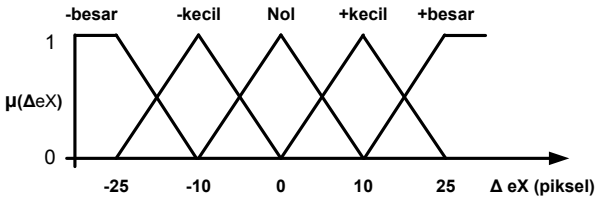
Gambar 3.19. Diagram blok kontroler proses *auto docking* tahap kedua

(a). Meluruskan posisi (b). Meluruskan arah hadap

Pada FLC#1 *input membership function* yang digunakan ditunjukan pada Gambar 3.20. Sedangkan untuk *output membership function* ditunjukan pada Gambar 3.21. Rule yang digunakan pada FLC#1 mengacu pada Tabel 3.2.



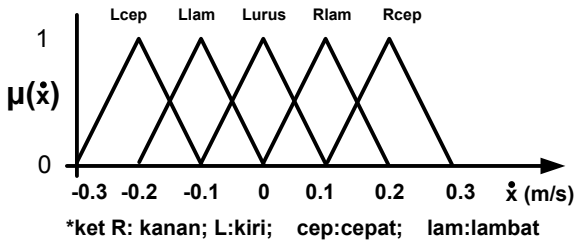
(a)



(b)

Gambar 3.20. Input membership function FLC#1

(a). Error X (b). Delta error X

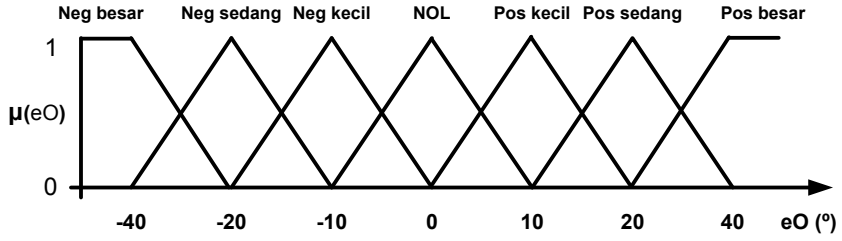


Gambar 3.21. Output membership function FLC#1

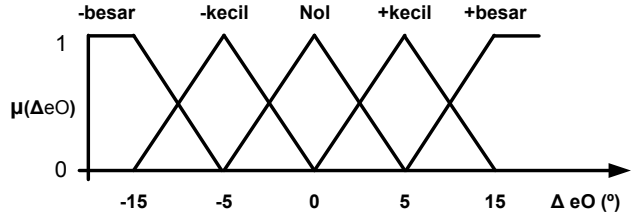
Tabel 3.2. Rule mendapatkan output $\dot{x}$

Rule #1	**eX**							
ΔeX		**Neg besar**	**Neg sedang**	**Neg kecil**	**Nol**	**Pos kecil**	**Pos sedang**	**Pos besar**
	-besar	Lcep	Llam	Lurus	Lurus	Lurus	Rlam	Rcep
	-kecil	Lcep	Lcep	Llam	Lurus	Rlam	Rcep	Rcep
	Nol	Lcep	Lcep	Llam	Lurus	Rlam	Rcep	Rcep
	+kecil	Lurus	Lurus	Llam	Lurus	Rlam	Lurus	Lurus
	+besar	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus

Pada FLC#2 input membership function yang digunakan ditunjukan pada Gambar 3.22. Sedangkan untuk output membership function ditunjukan pada Gambar 3.23. Aturan atau rule yang digunakan pada FLC#2 untuk mendapatkan kecepatan rotasi ($\dot{\theta}$) mengacu pada Tabel 3.3.



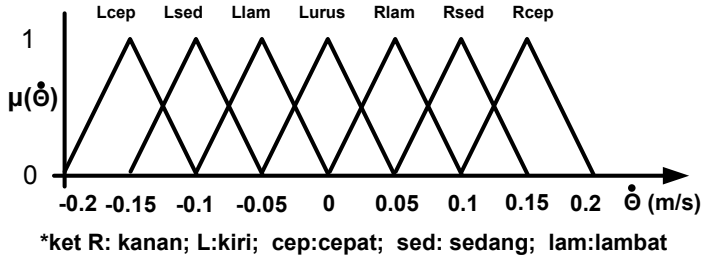
(a)



(b)

Gambar 3.22. Input membership function FLC#2

(a). Error O (b). Delta error O



Gambar 3.23. Output membership function FLC#2

Tabel 3.3. Rule mendapatkan output $\dot{\theta}$

Rule #2	**eO**							
ΔeO		**-besar**	**-sedang**	**-kecil**	**Nol**	**+kecil**	**+sedang**	**+besar**
	-besar	Rlam	Stop	Stop	Stop	Stop	Stop	Llam
	-kecil	Rsed	Rlam	Stop	Stop	Stop	Llam	Lsed
	Nol	Rcep	Rsed	Rlam	Stop	Llam	Lsed	Lcep
	+kecil	Rsed	Rlam	Stop	Stop	Stop	Llam	Lsed
	+besar	Rlam	Stop	Stop	Stop	Stop	Stop	Llam

Strategi *docking* pada tahap pertama dan kedua telah diselesaikan. Maka untuk melakukan penyambungan konektor hanya diperlukan satu tahap lagi yaitu tahap ketiga. Tahap ketiga ini dilakukan dengan cara menjalankan robot maju lurus ke depan. Robot akan berhenti saat telah mendeteksi penyambungan konektor. Tahap ketiga dilakukan tanpa adanya panduan arah pergerakan. Hal ini disebabkan jarak antara robot dan *docking station* terlalu dekat sehingga *depth* sensor pada kamera kinect tidak mampu membaca jarak.

Halaman ini sengaja dikosongkan

BAB 4

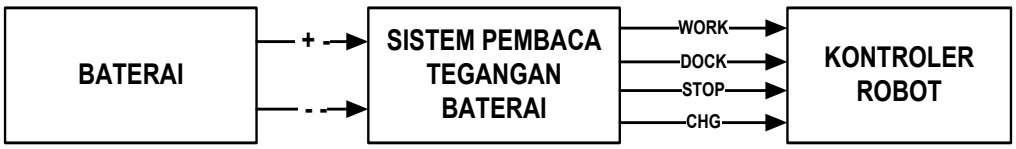
HASIL DAN PEMBAHASAN

Bab ini membahas tentang beberapa tahap pembuatan sistem *auto docking* pada *service robot* menggunakan persepsi visual. Pada bagian ini dibahas mengenai hasil pengujian sistem beserta analisa yang diperoleh dari data hasil percobaan. Pengujian yang dilakukan mencakup pengujian pembacaan tegangan baterai, pengujian sistem deteksi *docking station* hingga pengujian proses *auto docking* hingga robot berhasil menyambungkan konektor.

4.1. Pengujian Pembacaan Tegangan Baterai

Pengujian pembacaan tegangan baterai bertujuan untuk mengetahui keakuratan sistem dalam membaca tegangan baterai. Pengujian ini dilakukan dalam dua kondisi yaitu kondisi pengisian atau *charging* dan kondisi pengosongan atau *discharging*. Hasil dari pengujian ini adalah sinyal perintah ke kontroler robot untuk melakukan pengisian baterai atau melepas sambungan saat pengisian.

Pengujian ini dilakukan dengan menghubungkan baterai ke sistem pembaca tegangan baterai. Setelah terhubung dengan baterai, sistem selaanjutnya mulai membaca kondisi *charge/discharge* dan membaca tegangan baterai saat itu. Percobaan pertama dilakukan saat kondisi discharging dengan memberikan tegangan 24,4 volt hingga 23,3 volt dengan interval 0.1 volt. Percobaan kedua dilakukan saat kondisi charging dengan memberikan tegangan 24,4 volt hingga 26,5 volt dengan interval 0.1 volt. Hasil pengujian ditunjukan pada Tabel 4.1.



Gambar 4.1. Seting percobaan pembacaan tegangan baterai

Tabel 4.1. Pengujian sistem pembaca tegangan baterai

No	Tegangan Baterai (V)	*Charge*/ *Discharge*	Sinyal Kontrol
1	24,4	*Discharge*	WORK
2	24,3	*Discharge*	WORK
3	24,2	*Discharge*	WORK
4	24,1	*Discharge*	WORK
5	24	*Discharge*	WORK
6	23,9	*Discharge*	WORK
7	23,8	*Discharge*	WORK
8	23,7	*Discharge*	WORK
9	23,6	*Discharge*	WORK
10	23,5	*Discharge*	DOCK
11	23,4	*Discharge*	DOCK
12	23,3	*Discharge*	DOCK
13	24,6	*Charge*	CHG
14	24,7	*Charge*	CHG
15	24,8	*Charge*	CHG
16	24,9	*Charge*	CHG

No	Tegangan Baterai (V)	*Charge*/ *Discharge*	Sinyal Kontrol
17	25	*Charge*	CHG
18	25,1	*Charge*	CHG
19	25,2	*Charge*	CHG
20	25,3	*Charge*	CHG
21	25,4	*Charge*	CHG
22	25,5	*Charge*	CHG
23	25,6	*Charge*	CHG
24	25,7	*Charge*	CHG
25	25,8	*Charge*	CHG
26	25,9	*Charge*	CHG
27	26	*Charge*	CHG
28	26,1	*Charge*	CHG
29	26,2	*Charge*	CHG
30	26,3	*Charge*	CHG
31	26,4	*Charge*	STOP
32	26,5	*Charge*	STOP

Sinyal kontrol yang dihasilkan oleh sistem pembaca tegangan baterai ada 4 yaitu DOCK yang menunjukan bahwa saatnya robot melakukan pengisian baterai. WORK yang menunjukan bahwa robot harus tetap berkerja. CHG yang menunjukan proses pengisian sedang berlangsung, dan STOP yang menunjukan proses pengisian baterai dihentikan. Lama pengisian hingga mencapai tegangan maksimal adalah 135 menit.

4.2. Pengujian Sistem Pendeteksi *Docking Station*

4.2.1. Pengujian Dengan Variasi Jumlah Bin Histogram

Sistem pendeteksi *docking station* yang digunakan pada robot, dibuat menggunakan *histogram matching*. Histogram tersebut memiliki jumlah bin sebanyak 256 buah yang mewakili pola-pola LBP yang ada pada gambar. Namun dalam proses *histogram matching*, tidak semua bin diproses. Proses perhitungan hanya dilakukan dengan beberapa bin yang memiliki nilai dominan. Hal tersebut dilakukan untuk mempersingkat waktu komputasi.

Tabel 4.2. Hasil pengujian dengan variasi jumlah bin

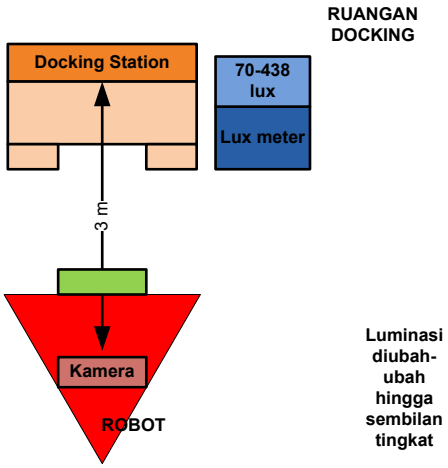
Jumlah bin	Akurasi (%)	Waktu komputasi (ms)
1	2	115
2	60	118
3	97	122
4	100	124
5	100	126
6	100	128
7	100	130
8	82	131
9	17	133
10	5	136
11	3	138
12	2	141

Pengujian ini bertujuan untuk mengetahui jumlah bin yang optimal yaitu jumlah bin mampu menghasilkan akurasi tertinggi dengan waktu komputasi tercepat. Pendeteksian dilakukan pada jarak 3 meter dengan orientasi *docking station* 0°. Jumlah bin yang diujikan mulai dari 1 hingga 12 dengan interval 1. Hasil pengujian yang dilakukan dapat dilihat pada Tabel 4.2.

Dari pengujian yang dilakukan dapat dilihat bahwa akurasi tertinggi dengan waktu komputasi tercepat terjadi saat jumlah bin yang digunakan adalah 4 bin. Dengan jumlah bin 4 akurasi yang dihasilkan mencapai 100% dengan waktu komputsi 124 ms. Dengan hasil pengujian ini maka dalam menerapkan sistem pendeteksi *docking station* pada robot akan digunakan jumlah bin 4.

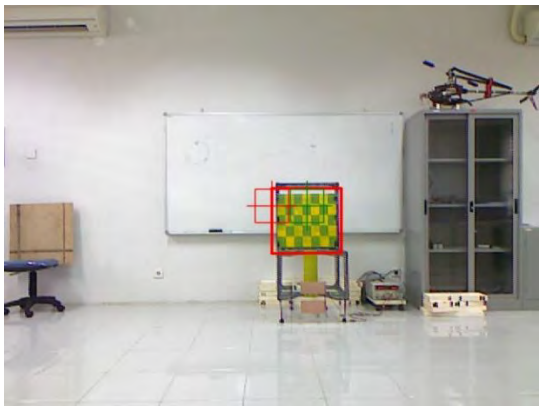
4.2.2. Pengujian Pada Variasi Luminasi Cahaya

Mengetahui lokasi *docking station* merupakan syarat utama sebelum melakukan proses *docking*. Jika lokasi *docking station* gagal ditemukan maka robot tidak akan bisa melakukan pengisian baterai. Maka untuk mengetahui lokasi *docking station* dalam penelitian ini digunakan sistem pendeteksi obyek atau object detection. Sistem yang dibangun harus mampu mendeteksi *docking station* dalam kondisi luminasi yang berubah-ubah. Walaupun *docking station* berada dalam ruangan namun menjaga luminasi selalu konstan agak sulit dilakukan. Gangguan sinar matahari melalui jendela dapat mengganggu kinerja sistem pendeteksi *docking station*.



Gambar 4.2. Seting percobaan pada perubahan luminasi

Kondisi lingkungan sekitar *docking station* mempengaruhi performa robot dalam melakukan pendeteksian *docking station*. Perubahan luminasi adalah salah satu faktor penyebab baik atau tidaknya performa suatu sistem pendeteksi obyek. Dalam pengujian ini pendeteksian *docking station* dilakukan pada sembilan level luminasi. Tingkat luminasi paling rendah yang digunakan adalah 75-78 lux sedangkan luminasi tertinggi adalah 445-451 lux. Pengukuran luminasi dilakukan dengan menggunakan lux meter dekko ft-7932 dengan seting percobaan ditunjukan pada Gambar 4.2. Luminasi diubah-ubah dengan memanfaatkan lampu dan mengatur cahaya matahari yang masuk ke ruangan. Saat *docking station* terdeteksi, tampilan program yang dijalankan ditunjukan dengan Gambar 4.3.



Gambar 4.3. Tampilan program saat sistem mendeteksi *docking station*

Hasil pengujian akurasi terhadap perubahan luminasi ditampilkan pada Tabel 4.3. Pengujian diawali dengan tingkat luminasi 75-78 lux. Pada tingkat ini kondisi ruangan cukup gelap. Dari 300 sampel gambar yang diujikan hanya terdeteksi sebanyak 196 buah gambar. Pada tingkat luminasi ini sistem memiliki akurasi yang sangat rendah. Pada lingkungan yang gelap citra yang diperoleh memiliki intensitas yang rendah dan antar piksel memiliki kontras yang kecil.

Pada tingkat luminasi 116-118 lux akurasi sistem mencapai 91.2%. Dari 300 sampel yang diujikan terdapat 279 gambar yang terdeteksi. Pada luminasi ini gambar yang dihasilkan kamera memiliki kontras yang baik. Dengan kontras antar piksel yang baik maka proses percarian nilai LBP semakin optimal. Akurasi tertinggi didapatkan pada tingkat luminasi 235-238 lux yaitu dengan akurasi 96.6%.

Dengan meningkatnya tingkat luminasi, akurasi sistem perlahan mulai turun. Namun hingga tingkat luminasi 391-395 lux akurasi sistem dinilai masih cukup baik karena memiliki akurasi diatas 80%. Pada luminasi diatas 395 lux akurasi sistem turun hingga dibawah 80%. Pada luminasi 445-451 lux akurasi sistem hanya 76.6%. Tingkat akurasi yang menurun disebabkan oleh permasalahan yang sama dengan saat luminasi rendah. Permasalahan tersebut adalah gambar hasil capture kamera memiliki kontras antar piksel yang rendah.

Tabel 4.3. Hasil pengujian akurasi pada perubahan luminasi

No	**Tingkat Luminasi (lux)**	**Jumlah sampel**	**Jumlah yang terdeteksi**	**Akurasi (%)**
1	75-78	300	196	65,3
2	116-118	300	279	93
3	205-209	300	280	93,3
4	235-238	300	290	96,6
5	281-285	300	275	91,6
6	315-318	300	270	90
7	351-356	300	265	88,3
8	391-395	300	257	85,6
9	445-451	300	230	76,6
Jumlah		2700	2342	**86,7**

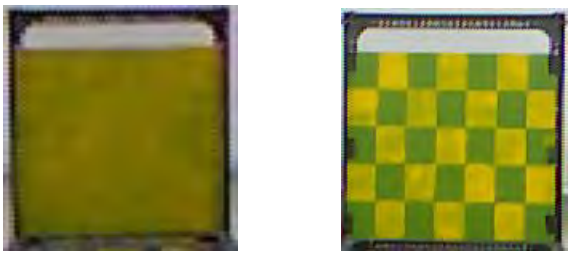
4.2.3. Pengujian Dengan Gangguan Obyek Lain

Dalam melakukan pendeksian *docking station*, robot tidak hanya menangkap gambar *docking station* saja. Kemungkinan adanya benda yang memiliki warna dan pola yang menyerupai *docking station* sangat besar. Maka dalam pengujian ini dilakukan pencarian benda-benda yang menyerupai *docking station* baik bentuk maupun warnanya. Pengujian ini dilakukan untuk mengetahui ketahanan sistem saat ada obyek lain dalam satu frame atau gambar.

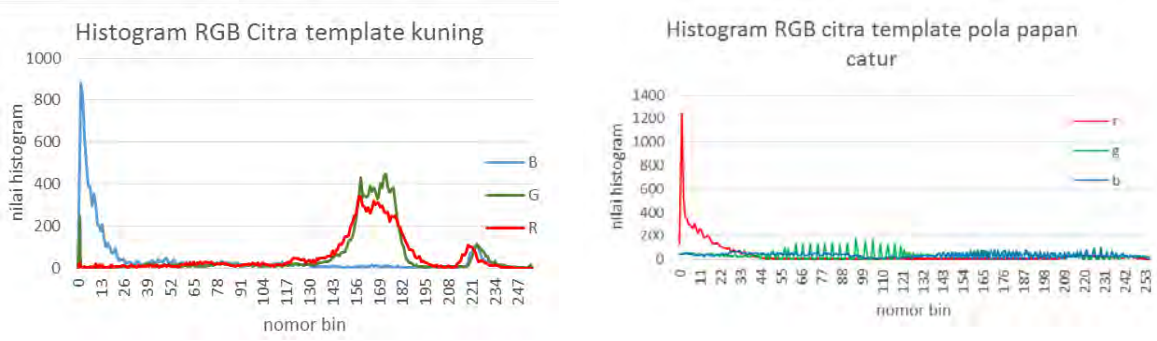
4.2.3.1.Dengan *Docking Station* Yang Berbeda Warna

Docking station yang digunakan dalam penelitian ini memiliki target area dengan pola papan catur. Namun *docking station* dideteksi bukan berdasarkan warna RGB nya melainkan dari LBP yang didapatkan dari konversi citra *grayscale*. Sehingga warna tersebut bisa diganti dengan warna lain asalkan memiliki nilai LBP yang sama nantinya. Dalam pengujian ini *docking station* yang berpola papan catur akan dibandingkan dengan *docking station* yang warna target areanya diganti menjadi kuning.

Dalam pengujian ini ada beberapa variabel yang diamati yaitu histogram RGB, histogram *grayscale* dan histogram LBP dari kedua warna *docking station*. Citra *template* dari masing-masing warna *docking station* dapat dilihat pada Gambar 4.4. Kedua citra tersebut lalu di proses hingga didapatkan nilai histogram RGB, *grayscale*, dan LBP. Pada obyek yang berbeda warna maka akan menunjukan hasil histogram LBP yang berbeda.



Gambar 4.4. Citra *template docking station*



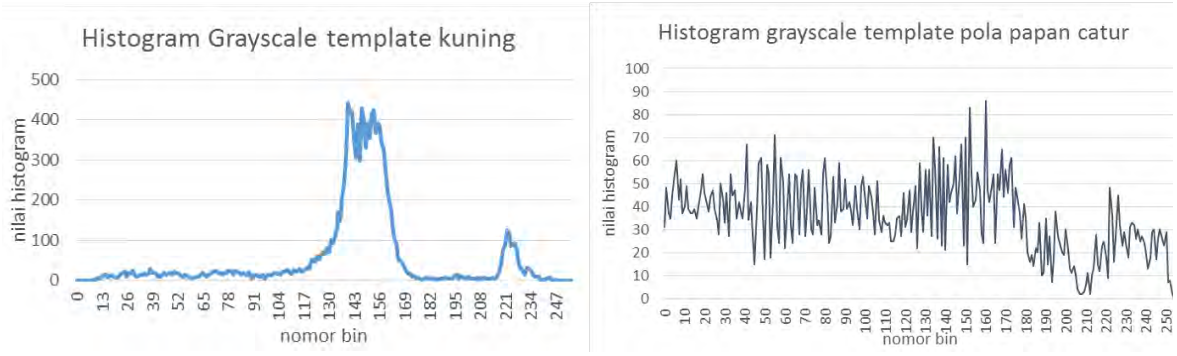
(a) (b)

Gambar 4.5. Histogram RGB dari Citra *template*
(a). Warna kuning (b). Pola papan catur

Hasil histogram RGB pada masing-masing citra *template* ditunjukan pada Gambar 4.5. Gambar 4.5(a) menunjukan histogram RGB citra *template* warna kuning. Dari histogram yang ditunjukan pada Gambar 4.5(a), warna biru lebih dominan pada saat intensitas rendah yaitu dari 0 hingga 24. Berbeda dengan Gambar 4.5(b) yang menunjukan warna merah dominan pada intensitas 0 hingga 24. Hal yang sama berlaku pada warna biru dan hijau yang memiliki intensitas dominan pada nilai yang berbeda.

Langkah pengujian berikutnya adalah melakukan konversi pada kedua citra *template* tersebut menjadi *grayscale*. Tujuan proses ini adalah untuk mengetahui apakah setelah dikonversi menjadi *grayscale*, kedua citra *template* tetap menunjukan hasil histogram yang berbeda atau tidak. Meskipun proses deteksi *docking station* menggunakan LBP tetapi citra *grayscale* perlu diamati karena merupakan langkah terakhir sebelum di konversi menjadi LBP. Hasil histogram *grayscale* dari citra *template* ditunjukan pada Gambar 4.6.

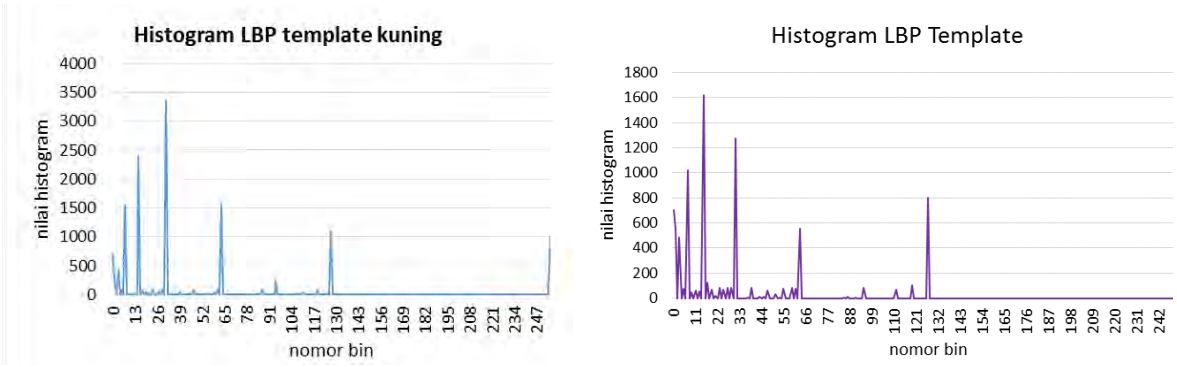
Setelah kedua gambar menjadi *grayscale* maka langkah berikutnya adalah melakukan konversi menjadi LBP. Sama seperti langkah sebelumnya, pada langkah ini juga akan diamati histogram LBP dari kedua gambar. Kedua histogram LBP kemudian dicari nilai kesamaanya menggunakan histogram intersection. Histrogram LBP kedua gambar dapat dilihat pada Gambar 4.7.



(a) (b)

Gambar 4.6. Histogram *grayscale* dari Citra *template*

(a). Warna kuning (b). Pola papan catur



(a) (b)

Gambar 4.7. Histogram LBP dari Citra *template*

(a). Warna kuning (b). Pola papan catur

Dari Gambar 4.7, dapat dilihat secara langsung bahwa kedua histogram memiliki pola yang berbeda. Pada histogram *template* warna kuning, 4 pola dominan tertinggi masing-masing ditunjukan oleh bin 31, 15, 63, dan 7. Sedangkan pada *template* pola papan catur 4 pola dominan tertinggi masing-masing ditunjukan oleh bin 15, 31, 7, dan 127. Dari persamaan histogram intersection didapatkan nilai kesamaan histogram sebesar 17%. Nilai kesamaan tersebut sangat rendah sehingga dapat disimpulkan bahwa *docking* pola papan catur tidak bisa langsung digunakan menggunakan *template docking* berwarna kuning.

4.2.3.2.Dengan Obyek Yang Menyerupai *Docking Station*

Di dalam pengujian di lab riset pusat robotika its ada beberapa benda yang mengganggu pendeteksian *docking station*. Benda-benda tersebut adalah sebuah lemari besi, dan pembatas meja. Dalam pengujian juga akan diberikan gangguan dengan sebuah benda yang dibentuk menyerupai *docking station*. Benda tersebut berupa kursi yang diatasnya diberi kayu untuk target areanya. Benda-benda yang diamati dalam pengujian ini dapat dilihat pada Gambar 4.8.

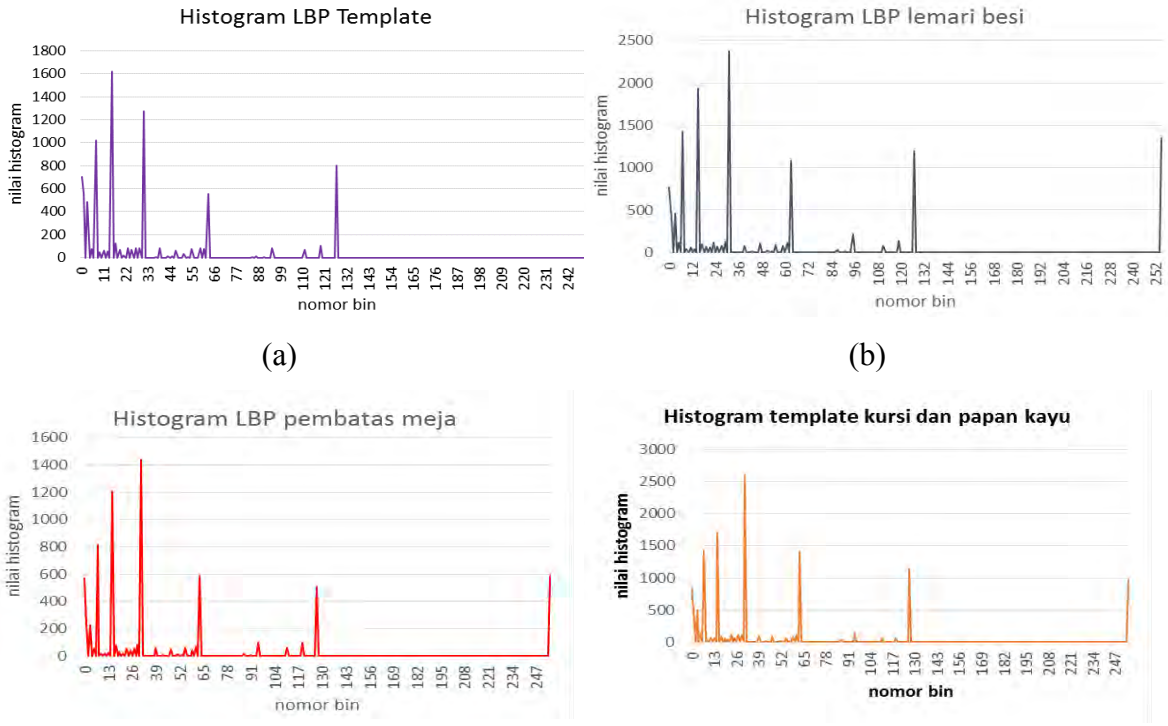
Ketiga benda tersebut terdeteksi oleh sistem sebagai *docking station* padahal benda tersebut bukanlah *docking station*. Benda pada Gambar 4.8(a) dan (b) merupakan benda yang berada pada lab tempat pengujian. Sedangkan benda pada Gambar 4.8(c) dibuat untuk menguji keberhasilan sistem dalam membedakan benda yang meyerupai *docking station*.

Jika dilihat secara langsung Gambar 4.8(a) dan (b) memiliki warna yang berbeda dengan *docking station*. Kedua benda tersebut berwarna abu-abu dan biru sedangkan *docking station* memiliki pola papan catur. Jadi dalam pembahasan ini akan diamati histogram LBP dari kedua benda tersebut. Sistem pendeteksi obyek yang diterapkan menggunakan LBP sehingga kemiripan histogram antara *docking station* dengan kedua benda tersebut yang menyebabkan kesalahan deteksi. Gambar 4.9 menunjukan histogram LBP pada masing-masing obyek yang diamati.

(a) (b) (c)

Gambar 4.8. Benda-benda yang terdeteksi sebagai *docking station*

(a). Lemari besi (b). Pembatas meja (c). Kursi dengan target area berupa kayu

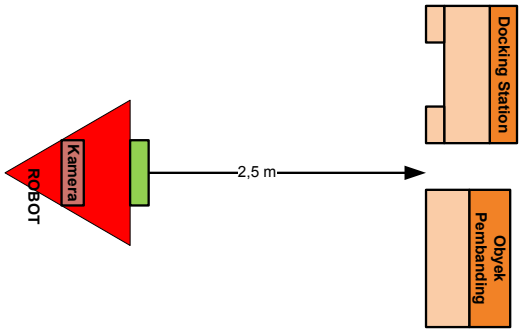


Gambar 4.9. Histogram LBP dari obyek yang menyerupai *docking*

(a). *Docking station* (b). Lemari besi (c). Pembatas meja (d). Kursi dan kayu

Desain sistem pendeteksi yang diterapkan menggunakan 4 buah bin dominan yang dimasukan ke proses *histogram matching*. Dari histogram LBP *docking station*, 4 bin yang dominan adalah bin nomor 15, 31, 7, dan 127. Pada obyek lainya nomor bin 31 dan 15 merupakan 2 bin dominan kecuali bin dominan ketiga dan ke empat yaitu bin nomor 7 dan 63. Pengujian dengan sebuah kursi dengan target area berupa kayu dilakukan untuk mengetahui akurasi sistem dalam membedakan *docking station* asli dengan benda yang dibentuk menyerupai *docking station*.

Pengujian ini dilakukan dengan cara menampilkan *docking station* dan obyek yang menyerupai pada satu gambar. Hasil pengujian ini adalah tingkat akurasi sistem dalam mengenali *docking station* yang sebenarnya. Pengujian ini dilakukan dengan jarak pendeteksian 2,5 m. Seting percobaan dapat dilihat pada Gambar 4.10.



Gambar 4.10. Seting Percobaan dengan obyek pembanding

Tabel 4.4. Akurasi sistem saat ada obyek meyerupai *docking station*

No	Obyek pembanding	Jumlah gambar	Jumlah terdeteksi	Akurasi
1	Lemari besi	175	163	93,14%
2	Pembatas meja	175	171	97,71%
3	Kursi dengan kayu	175	152	86,82%

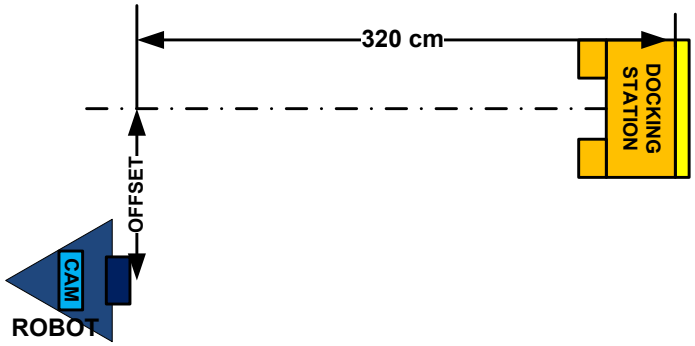
Tiap-tiap pengujian pada masing-masing obyek dilakukan sebanyak 175 gambar. Pengujian pertama dilakukan dengan lemari besi sebagai obyek pembanding. Selanjutnya menggunakan pembatas meja dan yang terakhir menggunakan kursi yang memakai target area berupa kayu. Hasil pengujian ditampilkan pada Tabel 4.4.

4.3. Pengujian Proses *Auto Docking*

4.3.1. Pengujian *Auto Docking* Tahap Pertama

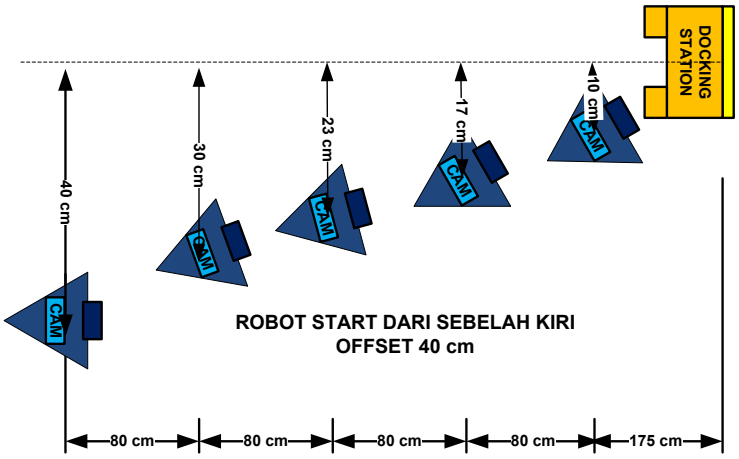
Pengujian tahap pertama ini dilakukan sebanyak tujuh kali percobaan. Pada setiap percobaan robot dijalankan pada jarak 320 cm di depan *docking station*. Namun setiap percobaan memiliki perbedaan jarak offset. Jarak offset ini merupakan selisih jarak yang dibentuk antara posisi robot saat *start* dengan arah depan dokcing station. Jarak offset yang diujikan adalah 0 cm, 40 cm, 80 cm, dan 120 cm pada masing-masing sisi kiri dan kanan.

Kontroler yang diterapkan di pengujian tahap pertama ditunjukan pada Gambar 3.15. Seting percobaan ditunjukan oleh Gambar 4.11. Pengujian ini bertujuan untuk mengetahui proses pergerakan robot hingga berhenti di depan *docking station*. Selain itu juga diamati nilai X dan D pada tiap waktu dari start hingga robot berhenti.

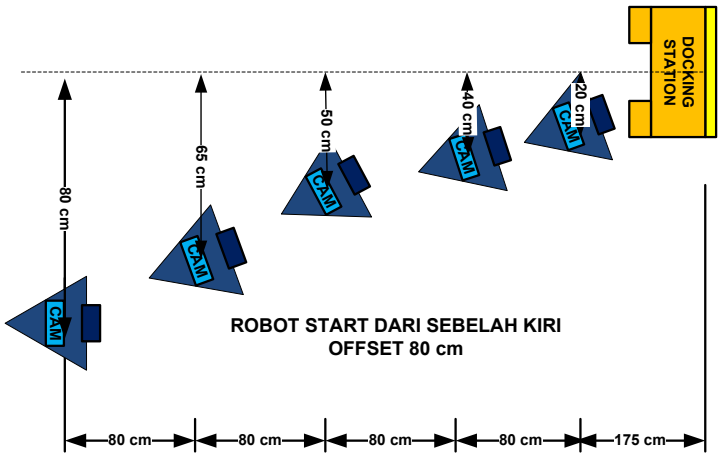


Gambar 4.11. Seting percobaan proses *docking* pada tahap pertama

Pengujian ini dimulai dengan posisi robot berada di sebelah kiri *docking station* dengan jarak offset 40 cm. Saat sistem dijalankan pertama-tama robot akan mendeteksi lokasi *docking station*. Dari hasil pendeteksian tersebut didapatkan nilai X dan D aktual. Saat nilai D dan X pada yang diperoleh berbeda dengan seting poin yang ditentukan maka robot akan berjalan mendekati *docking station*. Pergerakan yang dibentuk robot ditunjukan pada Gambar 4.12. Dari gambar tersebut dapat dilihat bahwa robot tidak selalu berhenti tepat di depan *docking station*. Sehingga memerlukan tahap dua dan tiga untuk proses penyambungan konektor.



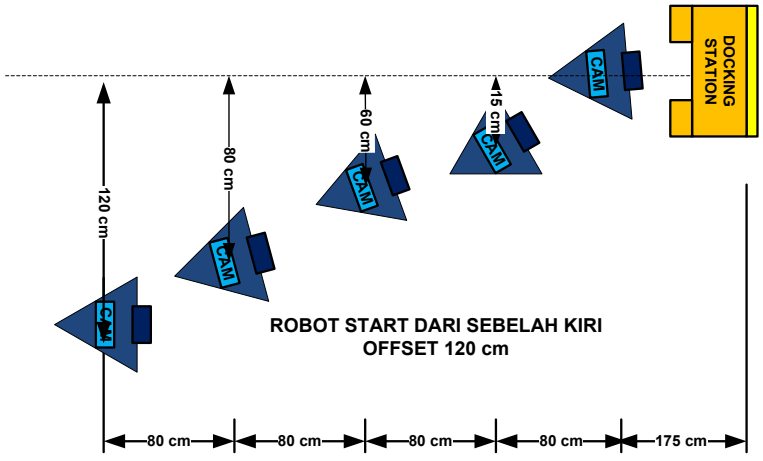
Gambar 4.12. Hasil percobaan tahap pertama posisi kiri dengan offset 40 cm



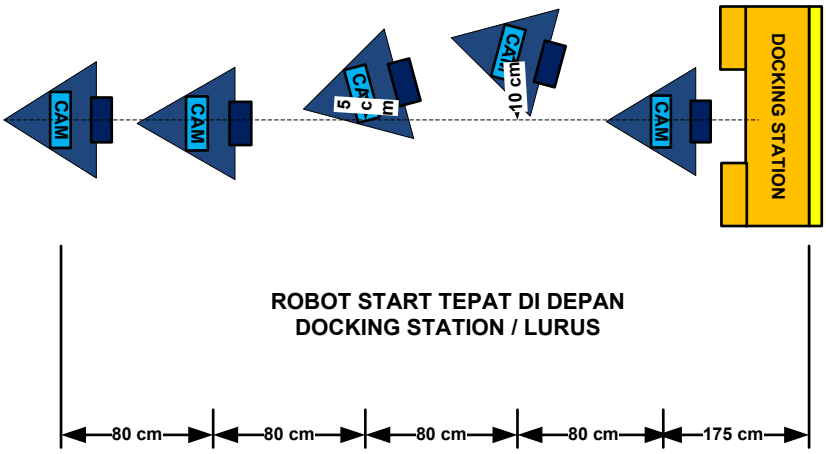
Gambar 4.13. Hasil percobaan tahap pertama posisi kiri dengan offset 80 cm

Pengujian dilanjutkan dengan mengubah jarak offset menjadi 80 cm. Dengan prosedur yang sama robot dijalankan sehingga membentuk pergerakan seperti yang ditunjukan pada Gambar 4.13. Pada percobaan ini robot berhenti dengan jarak 20 cm di sebelah kiri *docking station*. Dengan hasil tersebut masih menunjukan bahwa sistem masih membutuhkan tahap dua dan tiga.

Pengujian ketiga dilakukan dengan jarak offset 120 cm. Pergerakan robot ditunjukan pada Gambar 4.14. Pada pengujian ini robot berhenti tepat di depan *docking station* tetapi arah hadap robot tidak mengarah langsung ke *docking station*. Robot menghadap bagian kanan *docking station* sehingga proses penyambungan konektor tidak bisa langsung dilakukan.

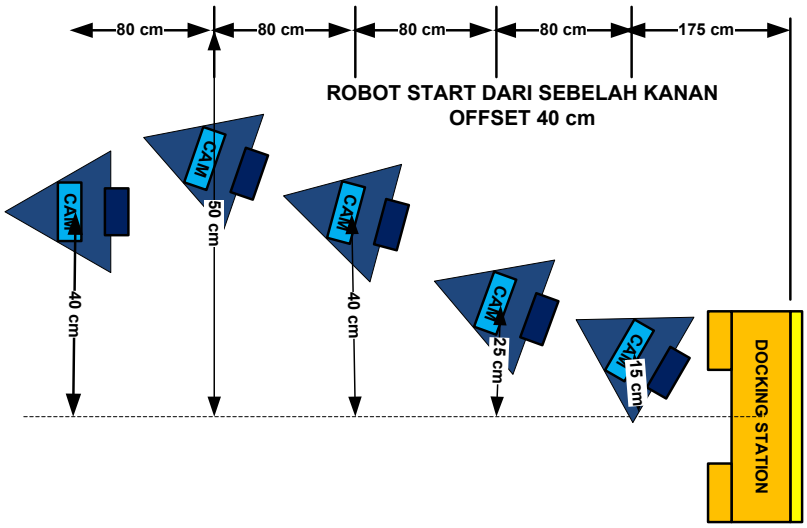


Gambar 4.14. Hasil percobaan tahap pertama posisi kiri dengan offset 120 cm



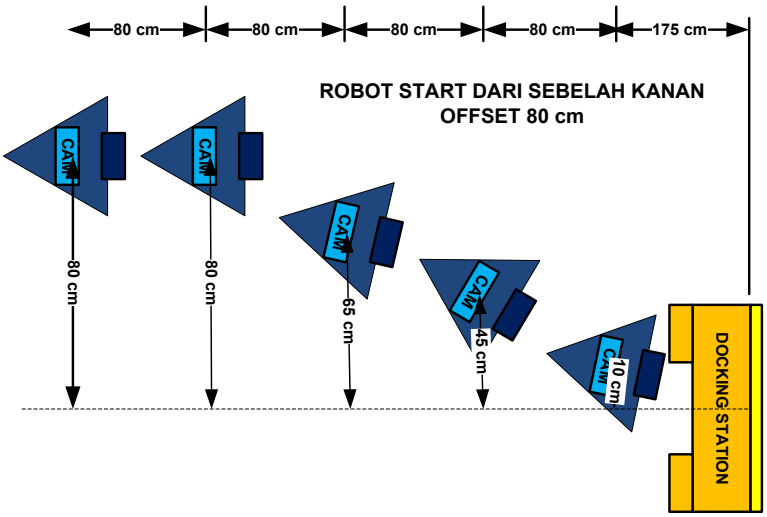
Gambar 4.15. Hasil percobaan tahap pertama posisi lurus

Pengujian dilanjutkan dengan jarak offset nol atau robot tepat di depan *docking station*. Pada posisi ini seharusnya robot mampu berjalan lurus dan berhenti tepat didepan *docking station*. Arah hadap yang robot seharusnya lurus menghadap *docking station*. Dari Gambar 4.15 yang menunjukan pergerakan robot. Pada jarak 240 cm dan 160 cm dari *docking station* robot sedikit terganggu sehingga bergeser hingga 10 cm. Tetapi saat berhenti robot mampu menghadap *docking station*.

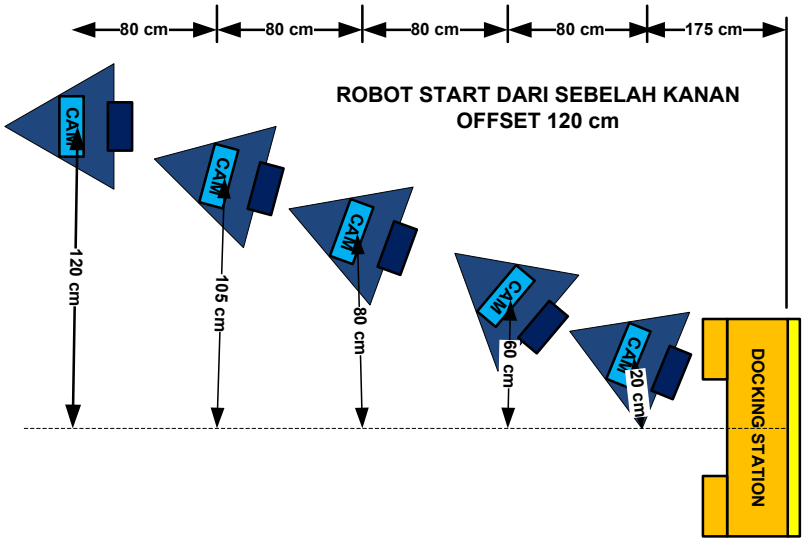


Gambar 4.16. Hasil percobaan tahap pertama posisi kanan dengan offset 40 cm

Posisi robot kemudian diubah ke sebelah kanan *docking station*. Dengan mengatur posisi offset pada jarak 40 cm kemudian robot dijalankan. Arah pergerakan robot ditunjukan dengan Gambar 4.16. Pada pengujian ini robot berhenti di sebelah kanan *docking station* dengan jarak 15 cm. Arah pergerakan robot pada pengujian dengan offset 80 cm dan 120 cm masing-masing ditunjukan Gambar 4.17 dan 4.18.



Gambar 4.17. Hasil percobaan tahap pertama posisi kiri dengan offset 80 cm



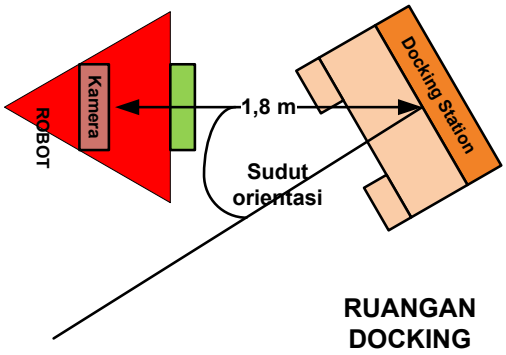
Gambar 4.18. Hasil percobaan tahap pertama posisi kiri dengan offset 120 cm

Dari seluruh pengujian pada tahap pertama ini menunjukan bahwa robot tidak selalu dapat berhenti dengan posisi saling berhadapan dengan *docking station*. Pembagian sistem *auto docking* menjadi beberapa tahap dinilai akan mempermudah robot melakukan penyambungan konektor. Setelah dilakukan pengujian tahap pertama kemudian dilanjutkan pengujian tahap kedua.

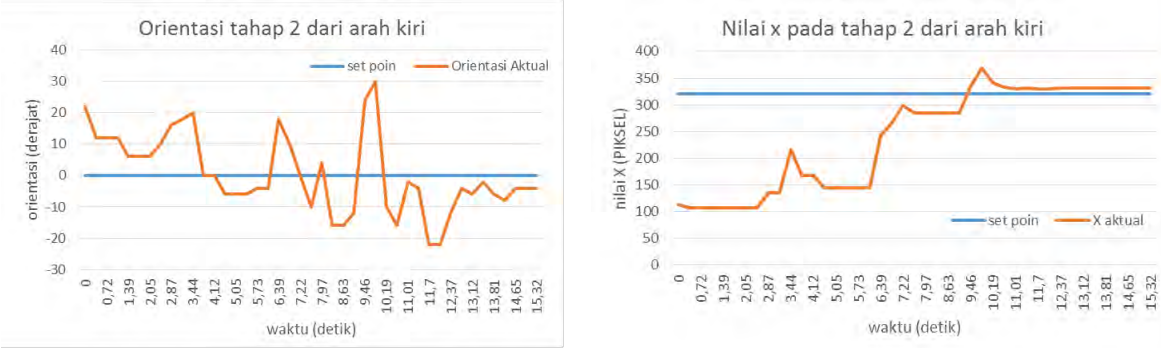
4.3.2. Pengujian *Auto Docking* Tahap Kedua

Tahap kedua mengacu pada nilai center piksel (X) dan orientasi (O) untuk dijadikan masukan kontroler. Kedua nilai tersebut diproses dalam sebuah *fuzzy logic controller* untuk menentukan arah pergerakan robot. Diagram blok kontroler pada tahap dua ditunjukan pada Gambar 3.19. Pengujian ini bertujuan untuk mengetahui proses pergerakan robot hingga dapat saling berhadapan dengan *docking station*. Selain itu juga diamati nilai X dan O pada tiap waktu dari awal hingga robot berhenti. Pada tahap ini dilakukan 3 kali percobaan yaitu saat robot berada di depan *docking station*, di sebelah kiri dan di sebelah kana *docking station*.

Pengujian pertama dilakukan dengan menjalankan robot di sebelah kiri *docking station*. Pada pengujian ini diamati arah pergerakan robot dalam meluruskan posisi agar saling berhadapan dengan *docking station*. Set poin masukan X adalah 320 sama dengan tahap pertama sedangkan set poin O adalah 0°. Saat kedua kondisi terpenuhi maka dipastikan robot dan *docking station* saling berhadapan. Seting percobaan yang dilakukan ditunjukan pada Gambar 4.19 berikut ini.



Gambar 4.19. Seting percobaan pada tahap kedua

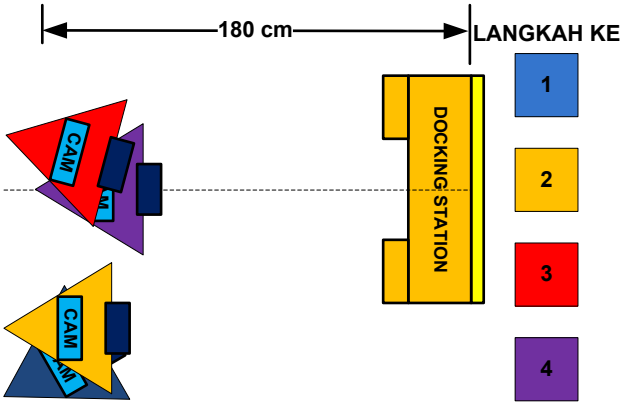


(a) (b)

Gambar 4.20. Nilai O dan X saat tahap dua dijalankan dari arah kiri
(a). Nilai O terhadap waktu (b). Nilai X terhadap waktu

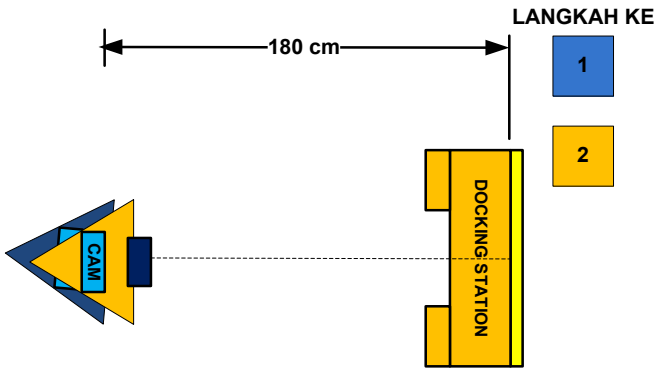
Sebelum robot mulai menjalankan tahap dua, pertama-tama robot membaca orientasi *docking station*. Pada percobaan ini orientasi awal yang terbaca adalah 22° sehingga robot menyesuaiakan arah hadapnya. Dalam proses penyesuaian ini robot melakukan pergerakan rotasi atau berputar di tempat. Penyesuaian ini berhenti saat robot hasil deteksi menunjukan orientasi bernilai 0°. Namun karena kontroler yang digunakan adalah *fuzzy logic controller* maka saat orientasi mendekati 0° proses penyesuaian orientasi dianggap selesai. Hal ini disebabkan konversi nilai crisp menjadi fuzzy memiliki rentang nilai tertentu misalnya antara -5° sampai dengan 5°. Agar robot dan *docking station* bisa saling berhadapan maka selain orientasi yang harus bernilai nol, nilai error X juga harus bernilai nol. Proses penyesuaian variabel X dilakukan saat nilai orientasi nol. Setelah kedua kondisi terpenuhi maka proses *auto docking* tahap dua telah terselesaikan.

Arah gerakan robot diilustrasikan pada Gambar 4.21. Pergerakan robot diawali dengan gambar nomor satu berwarna biru. Dari gambar tersebut menunjukan bahwa untuk memposisikan agar saling berhadapan, robot mengawali pergerakan dengan gerakan rotasi. Hasil pergerakan itu ditunjukan dengan gambar nomor dua berwarna kuning. Setelah melakukan gerakan rotasi dilanjutkan dengan pergerakan ke arah kiri atau kanan untuk memposisikan di depan *docking station*.

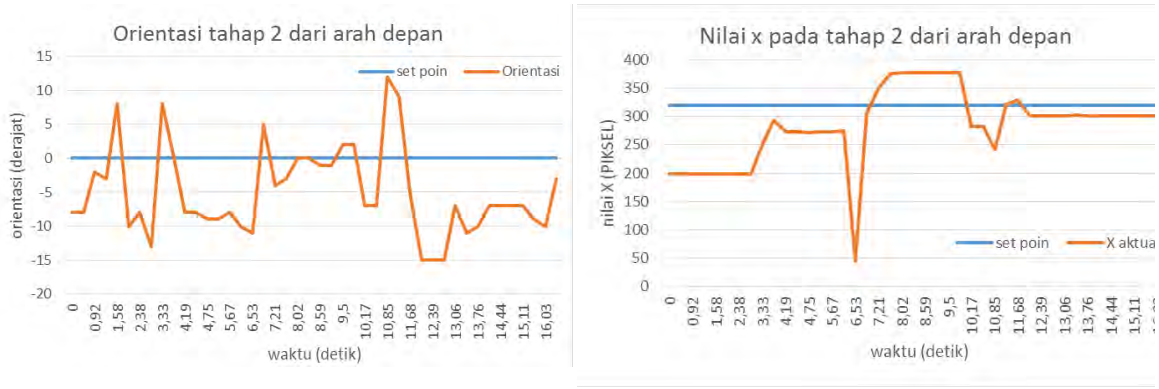


Gambar 4.21. Pergerakan robot pada tahap kedua dari arah kiri

Pengujian kemudian dilanjutkan dengan orientasi awal 0 atau lurus dari depan. Jarak antara *docking station* dengan robot tetap 180 cm. Seting percobaan dan kontroler yang diterapkan sama dengan pengujian pertama. Hasil pergerakan robot ditunjukan pada Gambar 4.22 sedangkan perubahan nilai orientasi dan nilai X dapat diamati pada Gambar 4.23. Saat robot dijalankan pada posisi depan *docking station* maka robot hanya melakukan pergerakan rotasi. Namun pembacaan orientasi menunjukan perbedaan hasil dimana posisi awal yang di set 0° menunjukan nilai -8°. Maka robot menjalankan pergerakan rotasi hingga membaca nilai orientasi mendekati 0°. Dalam percobaan ini robot berhenti saat membaca orientasi -3°. Setelah berhenti melakukan pegerakan rotasi robot seharusnya melakukan penyesuaian nilai X dengan bergerak ke samping. Namun nilai X yang terbaca mendekati seting poin sehingga tahap dua dianggap telah selesai.



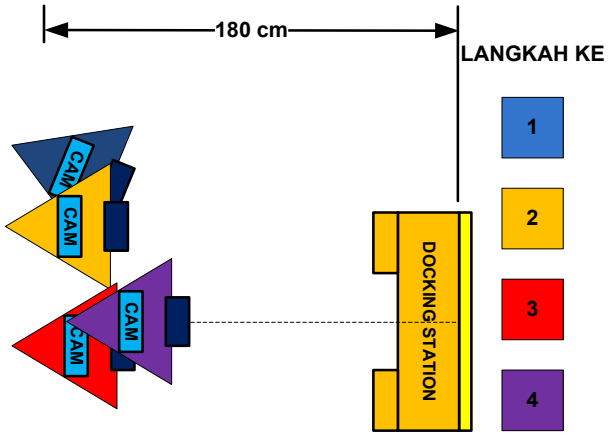
Gambar 4.22. Pergerakan robot pada tahap kedua dari arah depan



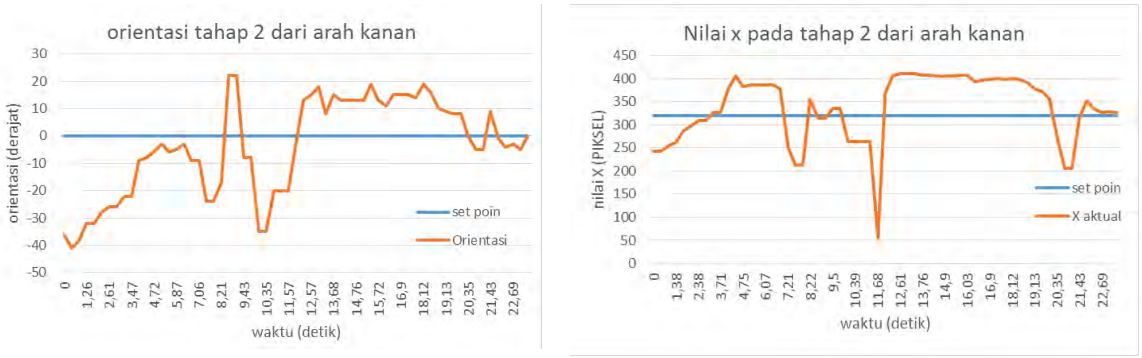
(a) (b)

Gambar 4.23. Nilai orientasi dan X saat tahap dua dijalankan dari arah depan (a). Nilai orientasi terhadap waktu (b). Nilai X terhadap waktu

Pengujian terakhir pada tahap dua dilakukan dengan memposisikan robot di sebelah kanan *docking station*. Dengan prosedur yang sama dengan percobaan sebelumnya, didapatkan hasil yang menunjukan pergerakan robot yang ditunjukan pada Gambar 4.24. Sedangkan hasil perubahan nilai X dan orientasi ditunjukan grafik pada Gambar 4.25. Pada percobaan ini robot berjalan pada posisi orientasi awal -35°. Robot kemudian melakukan pergerakan rotasi sehingga didapatkan nilai akhir orientasi 0°. Penyesuaian nilai X juga mendapatkan hasil akhir nol pada percobaan ini. Hal ini menunjukan bahwa robot dan *docking station* benar-benar saling berhadapan.



Gambar 4.24. Pergerakan robot pada tahap kedua dari arah kanan



(a) (b)

Gambar 4.25. Nilai orientasi dan X saat tahap dua dijalankan dari arah kanan (a). Nilai orientasi terhadap waktu (b). Nilai X terhadap waktu

Dari seluruh pengujian pada tahap kedua ini menunjukan bahwa robot mampu berhenti dengan posisi saling berhadapan dengan *docking station*. Namun untuk melakukan penyambungan konektor masih memerlukan tahap tiga. Namun karena posisi robot dan *docking station* sudah saling berhadapan maka pada tahap tiga akan dilaukan pergerakan maju ke depan hingga sensor mendeteksi penyambungan konektor. Setelah dilakukan pengujian tahap kedua kemudian dilanjutkan pengujian gabungan antara tahap pertama, kedua, dan ketiga.

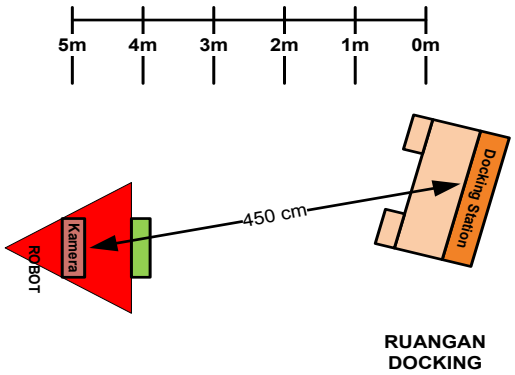
4.3.3. Pengujian *Auto Docking* Seluruh Tahap

Dalam pengujian ini robot dijalankan dari jarak 450 cm dari *docking station*. Akhir percobaan ini adalah saat robot melakukan penyambungan konektor. Seperti pengujian tahap sebelumnya, pada pengujian ini akan diamati proses pergerakan robot dan perubahan nilai X, D serta O pada tiap waktu. Kontroler yang digunakan sama dengan percobaan sebelumnya yaitu seperti pada Gambar 3.15 untuk tahap pertama dan Gambar 3.19 pada tahap kedua.

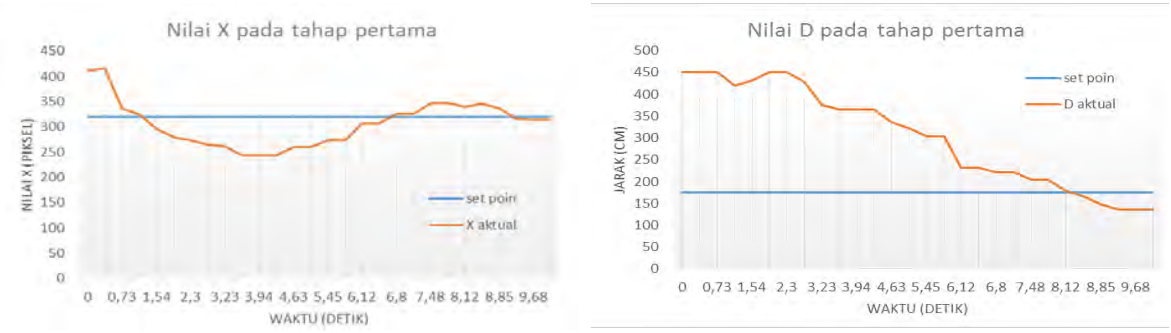
Pengujian dilakukan dengan seting percobaan yang ditunjukan pada Gambar 4.26. Pengujian ini dilakukan sebanyak 25 kali dengan keberhasilan penyambungan konektor 21 kali. Dari keseluruhan pengujian, sistem auto docking dapat dilakukan dengan rata-rata waktu 53.78 detik. Hasil keseluruhan pengujian proses auto docking ditunjukan pada Lampiran 2.A. Gambar 4.27 dan 4.28

merupakan salah satu hasil pengujian yang akan diamati hasil perubahan nilai X, D dan O. Hasil tersebut merupakan hasil pengujian tercepat yang didapatkan.

Pengamatan pertama dilakukan saat robot menjalankan tahap pertama. Pada tahap ini robot membutuhkan waktu selama 9,89 detik. Saat pertama dijalankan, robot membaca nilai X sebesar 412 dan nilai D sebesar 450 cm sehingga robot akan bergerak mendekati *docking station*. Nilai X dan D terus berubah hingga didapatkan nilai D dibawah set poin yaitu 175. Pada saat tersebut robot berhenti dan membaca nilai X dan orientasi untuk dilanjutkan ke tahap dua. Hasil perubahan nilai X dan D dapat dilihat dalam grafik pada Gambar 4.27.



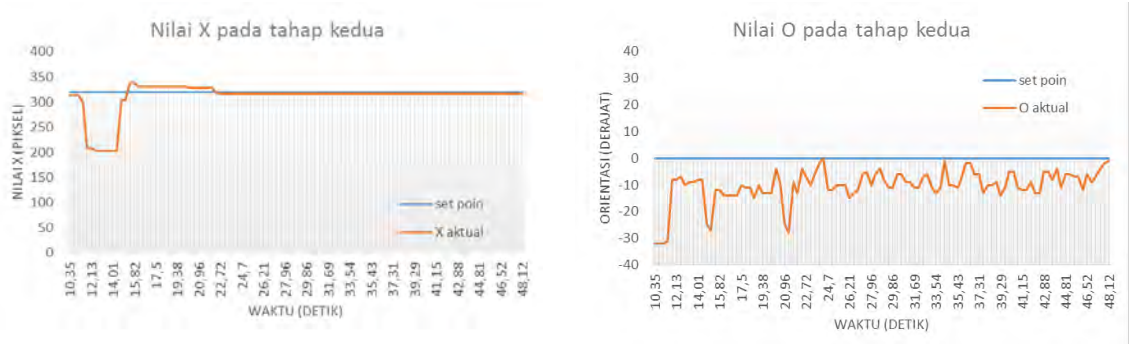
Gambar 4.26. Seting percobaan seluruh tahap



(a) (b)

Gambar 4.27. Perubahan nilai X dan D pada tahap pertama

(a). Nilai X terhadap waktu (b). Nilai D terhadap waktu



(a) (b)

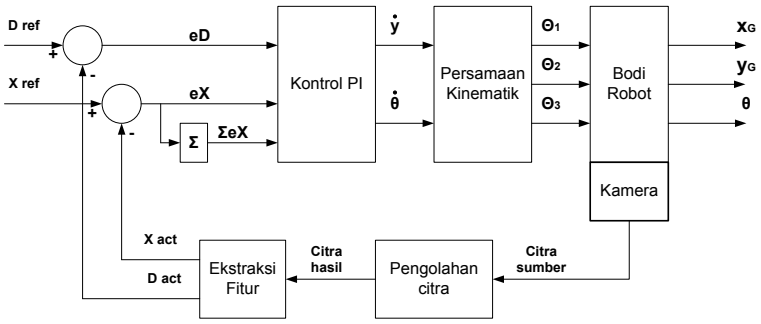
Gambar 4.28. Perubahan nilai X dan O pada tahap kedua
(a). Nilai X terhadap waktu (b). Nilai O terhadap waktu

Setelah melakukan tahap pertama robot melanjutkan pergerakan tahap dua. Nilai awal X dan Orientasi pada tahap dua masing-masing adalah 314 dan -32°. Karena mendeteksi nilai orientasi -32° maka robot melakukan pergerakan rotasi hingga orientasi mendekati nol. Dari gambar 4.30(a) dapat kita lihat bahwa nilai X setelah melewati detik ke 24 cenderung stabil mendekati set poin. Namun orientasi yang dibaca berubah ubah hingga bisa mendekati nol pada detik ke 46 lebih. Sehingga tahap dua selesai pada waktu detik ke 48,12. Selanjutnya proses tahap ketiga dilakukan dengan menjalankan robot lurus ke depan hingga terjadi penyambungan konektor. Tahap ketiga ini membutuhkan waktu 1,68 detik.

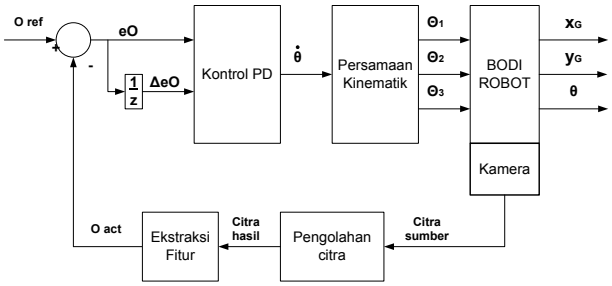
4.3.4. Pengujian Dengan Kontroler Konvensional

Penggunakan *fuzzy logic controller* didasari oleh kemudahan saat melakukan tuning. Berbeda dengan kontrol konvensional yang hanya berupa angka, fuzzy logic controler lebih mudah karena menggunakan bahasa linguistik. Selain itu faktor kesulitan menentukan model matematika dari robot juga menjadi alasan menggunakan *fuzzy logic controller*. Sebab *fuzzy logic controller* tidak memerlukan model matematika dari plan yang akan dikontrol. Dalam mendesain fuzzy yang diperlukan hanya pengetahuan pendesain tentang perilaku plan.

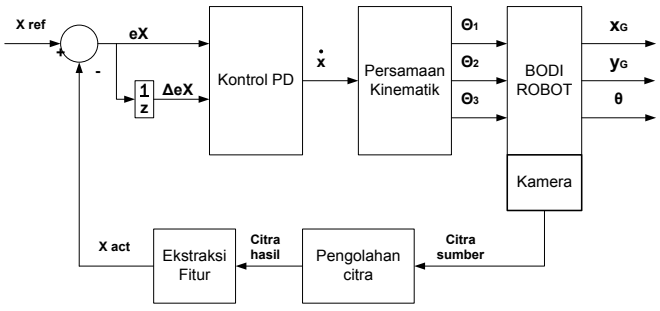
Namun untuk mengetahui apakah *fuzzy logic controller* benar-benar tepat digunakan sebagai kontroler, performanya harus dibandingkan dengan kontroler konvensional. Dalam pengujian ini kontroler konvensional yang digunakan adalah kontroler PI dan PD. Hal ini mengacu pada masukan kontroler fuzzi yang menggunakan masukan berupa error, integral error, dan delta error. Pengujian ini dilakukan dengan cara yang sama dengan pengujian sebelumnya. Kontroler yang digunakan dalam pengujian ini ditunjukan pada Gambar 4.29 dan Gambar 4.30.



Gambar 4.29. Diagram blok kontroler konvensional tahap pertama



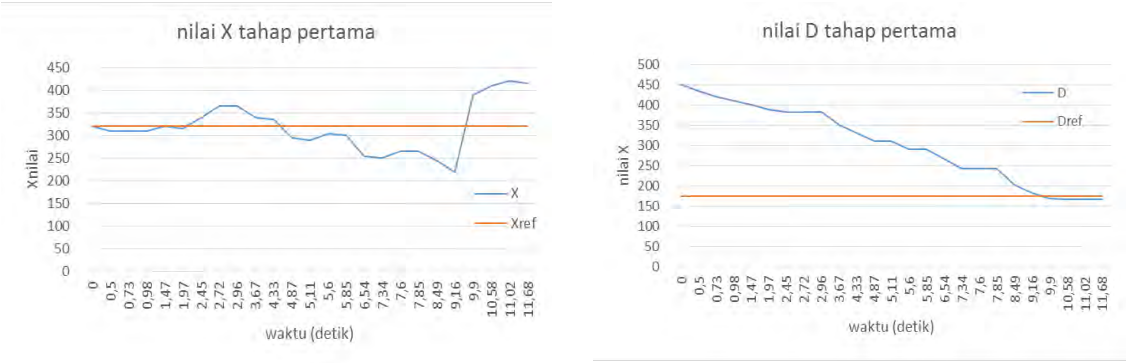
(a)



(b)

Gambar 4.30. Diagram blok kontroler proses *auto docking* tahap kedua

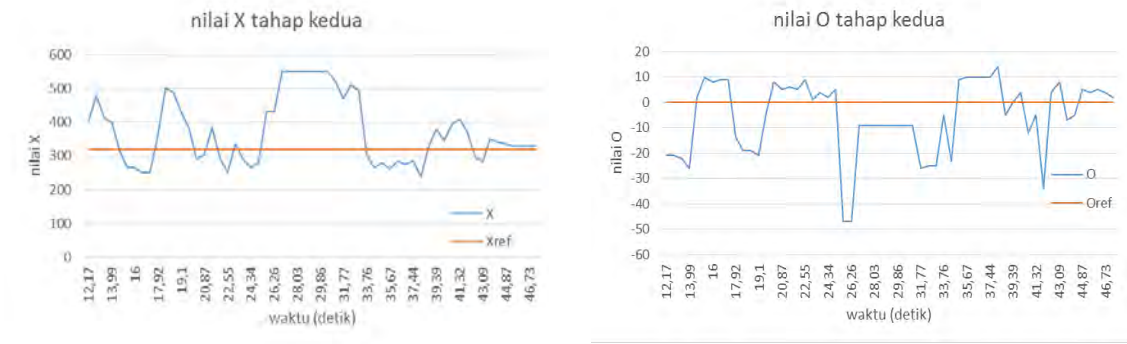
(a). Meluruskan posisi (b). Meluruskan arah hadap



(a) (b)

Gambar 4.31. Perubahan nilai X dan D tahap pertama dengan kontrol PI

(a). Nilai X terhadap waktu (b). Nilai D terhadap waktu



(a) (b)

Gambar 4.32. Perubahan nilai X dan O tahap kedua dengan kontrol PD

(a). Nilai X terhadap waktu (b). Nilai O terhadap waktu

Pengujian ini dilakukan sebanyak 20 kali dengan jarak pengujian 450 cm. Rata-rata waktu yang yang dibutuhkan untuk melakukan proses *auto docking* adalah 59,15 detik. Hasil ini menunjukan bahwa kontrol konvensional lebih lambat dari *fuzzy logic controller* pada pengujian sebelumnya. Dari 20 kali percobaan tersebut, robot mampu melakukan penyambungan konektor sebanyak 16 kali. Hasil pengujian ini ditunjukan pada Lampiran 2.B.

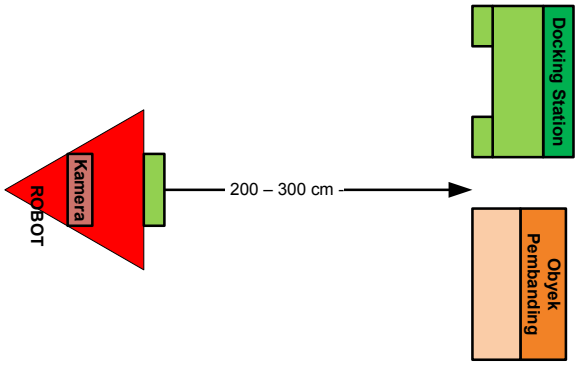
Dari hasil pengujian, proses *auto docking* tahap pertama diselesaikan selama 11,68 detik. Pada proses *auto docking* tersebut nilai X dan D yang

dijadikan masukan kontroler berubah hingga mendekati referensi. Pada saat tahap pertama berakhir, nilai X yang didapatkan adalah 415 dan nilai D yang didapatkan adalah 166 cm. Sehingga nilai error X dan D masing-masing adalah -95 dan 9 cm. Hasil perubahan nilai X dan D ditunjukan pada Gambar 4.31.

Setelah tahap pertama selesai, dilanjutkan proses tahap kedua. Tujuan tahap ini adalah didapatkanya nilai X dan O dengan nilai error mendekati nol. Pada akhir tahap kedua, nilai error X dan O yang didapatkan masing-masing sebesar -10 dan -2. Tahap ini diselesaikan dengan waktu 35,69 detik. Grafik perubahan nilai X dan O pada tahap kedua ditunjukan pada gambar 4.34. Proses selanjutnya adalah proses tahap ketiga untuk melakukan penyambungan konektor. Proses ini membutuhkan waktu 4.09 detik sehingga total waktu proses *auto docking* adalah selama 51,46 detik.

4.3.5. Pengujian *Auto Docking* Dengan Gangguan Obyek Lain

Dalam pengujian ini robot akan diganggu dengan sebuah benda yang menyerupai *docking station*. Pengujian ini bertujuan untuk mengetahui kemampuan robot dalam membedakan *docking station* dengan obyek yang menyerupai. Gangguan yang digunakan ditunjukan pada Gambar 4.8.(c). Dalam pengujian ini robot berhasil melakukan penyambungan konektor tanpa terganggu obyek lain. Seting percobaan ditunjukan pada Gambar 4.33.



Gambar 4.33. Pergerakan robot saat melakukan *auto docking* dengan gangguan

Pengujian ini dilakukan dengan menjalankan robot pada jarak 200 cm, 250 cm dan 300 cm dari *docking station*. Pada masing-masing jarak dilakukan pengujian sebanyak 8 kali sehingga ada 30 kali pengujian. Pada jarak 200 cm, dari 8 pengujian robot berhasil melakukan penyambungan konektor sebanyak 7 kali. Pada jarak 250 cm robot selalu berhasil melakukan penyambungan konektor. Pengujian terakhir dilakukan pada jarak 300 cm. Dalam pengujian ini robot sempat satu kali gagal melakukan penyambungan konektor.

Tabel 4.5. Hasil pengujian *auto docking* dengan gangguan

No	Jarak Pengujian	Jumlah Pengujian	Jumlah keberhasilan
1	200 cm	8	7
2	250 cm	8	8
3	300 cm	8	7

BAB 5

PENUTUP

Pada bagian penutup penelitian/tesis ini dibagi menjadi dua bagian, yaitu bagian kesimpulan dan saran. Bagian kesimpulan memuat rincian dan rangkuman hasil sistem *auto docking* pada *service robot* dengan persepsi visual. Sedangkan saran berupa beberapa kemungkinan yang dapat diaplikasikan untuk pengembangan penelitian lebih lanjut.

5.1. Kesimpulan

Robot berhasil menjalankan sistem *auto docking* menggunakan persepsi visual saat tegangan baterai dibawah 23.6 volt dan menyelesaikan pengisian baterai saat tegangan baterai diatas 26,4 volt. Dalam proses pengisian, tegangan baterai mencapai tegangan maksimal dalam waktu 135 menit.

Sistem *auto docking* menggunakan persepsi visual mampu bekerja dengan tingkat akurasi 86,7 % dan bekerja optimal pada rentang luminasi 116 lux hingga 395 lux. Sistem juga mampu membedakan antara *docking station* asli dengan obyek pembanding yang digunakan sebagai gangguan.

Robot melakukan penyambungan konektor dengan rata-rata waktu 53.78 detik dari jarak 450 cm dengan menggunakan *fuzzy logic controller*. Tingkat keberhasilan penyambungan konektor mencapai 84%. Hasil menggunakan *fuzzy logic controller* lebih cepat bila dibandingkan dengan kontroler konvensional yaitu 59.15 detik dari jarak 450 cm.

5.2. Saran

Berdasarkan hasil pengujian dan kesimpulan yang telah didapatkan, maka ada beberapa bagian yang dapat dikembangkan lebih lanjut. Pengembangan perlu dilakukan untuk mendapatkan sistem yang lebih akurat. Salah satunya adalah proses *histogram matching* dengan persamaan *histogram intersection* bisa

digantikan sebuah jaringan syaraf tiruan maupun kontrol cerdas lainya. Sistem kontrol yang diterapkan juga bisa dikembangkan agar robot bisa bergerak lebih cepat dengan akurasi penyambungan konektor yang lebih tinggi.

LAMPIRAN

LAMPIRAN 1

Rule untuk mencari nilai $\dot{\theta}$

	eX							
ΣeX	**eD= Jauh**	**Neg besar**	**Neg sedang**	**Neg kecil**	**Nol**	**Pos kecil**	**Pos sedang**	**Pos besar**
	Neg besar	Lscep	Lcep	Lsed	Llam	Llam	Lurus	Lurus
	Neg sedang	Lcep	Lsed	Llam	Llam	Lurus	Lurus	Rscep
	Neg kecil	Lcep	Lsed	Llam	Lurus	Rlam	Rlam	Rcep
	Nol	Lsed	Llam	Lurus	Lurus	Lurus	Rlam	Rsed
	Pos kecil	Lcep	Lsed	Llam	Lurus	Rlam	Rlam	Rcep
	Pos sedang	Lscep	Lurus	Lurus	Rlam	Rlam	Rsed	Rcep
	Pos besar	Lurus	Lurus	Rlam	Rlam	Rsed	Rcep	Rscep
	eX							
ΣeX	**eD= sedang**	**Neg besar**	**Neg sedang**	**Neg kecil**	**Nol**	**Pos kecil**	**Pos sedang**	**Pos besar**
	Neg besar	Lscep	Lcep	Lsed	Llam	Llam	Lurus	Lurus
	Neg sedang	Lcep	Lsed	Llam	Llam	Lurus	Lurus	Rscep
	Neg kecil	Lcep	Lsed	Llam	Lurus	Rlam	Rlam	Rcep
	Nol	Lsed	Llam	Lurus	Lurus	Lurus	Rlam	Rsed
	Pos kecil	Lcep	Lsed	Llam	Lurus	Rlam	Rlam	Rcep
	Pos sedang	Lscep	Lurus	Lurus	Rlam	Rlam	Rsed	Rcep
	Pos besar	Lurus	Lurus	Rlam	Rlam	Rsed	Rcep	Rscep
	eX							
ΣeX	**eD= dekat**	**Neg besar**	**Neg sedang**	**Neg kecil**	**Nol**	**Pos kecil**	**Pos sedang**	**Pos besar**
	Neg besar	Lscep	Lcep	Lcep	Llam	Llam	Lurus	Lurus
	Neg sedang	Lscep	Lsed	Lsed	Lurus	Lurus	Lurus	Rscep
	Neg kecil	Lcep	Lsed	Llam	Lurus	Rlam	Rlam	Rcep
	Nol	Lcep	Lsed	Lurus	Lurus	Lurus	Rlam	Rsed
	Pos kecil	Lcep	Lsed	Llam	Lurus	Rlam	Rlam	Rcep
	Pos sedang	Lscep	Lurus	Lurus	Lurus	Rlam	Rsed	Rscep
	Pos besar	Lurus	Lurus	Rlam	Rlam	Rcep	Rscep	Rscep
	eX							
ΣeX	**eD= sangat dekat**	**Neg besar**	**Neg sedang**	**Neg kecil**	**Nol**	**Pos kecil**	**Pos sedang**	**Pos besar**
	Neg besar	Lscep	Lscep	Lcep	Lurus	Llam	Lurus	Lurus
	Neg sedang	Lscep	Lcep	Lsed	Lurus	Lurus	Lurus	Rscep
	Neg kecil	Lscep	Lsed	Lsed	Lurus	Rlam	Rlam	Rscep
	Nol	Lscep	Lcep	Llam	Lurus	Rlam	Rcep	Rscep
	Pos kecil	Lscep	Lsed	Llam	Lurus	Rsed	Rcep	Rscep
	Pos sedang	Lscep	Lurus	Lurus	Lurus	Rsed	Rcep	Rscep
	Pos besar	Lurus	Lurus	Rlam	Lurus	Rcep	Rscep	Rscep

	eX							
ΣeX	eD= nol	Neg besar	Neg sedang	Neg kecil	Nol	Pos kecil	Pos sedang	Pos besar
	Neg besar	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus
	Neg sedang	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus
	Neg kecil	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus
	Nol	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus
	Pos kecil	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus
	Pos sedang	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus
	Pos besar	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus
	eX							
ΣeX	eD= melebihi dekat	Neg besar	Neg sedang	Neg kecil	Nol	Pos kecil	Pos sedang	Pos besar
	Neg besar	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus
	Neg sedang	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus
	Neg kecil	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus
	Nol	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus
	Pos kecil	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus
	Pos sedang	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus
	Pos besar	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus
	eX							
ΣeX	eD= melebihi jauh	Neg besar	Neg sedang	Neg kecil	Nol	Pos kecil	Pos sedang	Pos besar
	Neg besar	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus
	Neg sedang	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus
	Neg kecil	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus
	Nol	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus
	Pos kecil	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus
	Pos sedang	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus
	Pos besar	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus	Lurus

Rule untuk mencari nilai *ẏ*

eD						
Jauh	Sedang	Dekat	Sangat Dekat	Nol	Melebihi dekat	Melebihi jauh
Fcep	Fsed	Fsed	Flam	Nol	Rlam	Rsed

LAMPIRAN 2.A

Tabel pengujian auto docking seluruh tahap

No	Jarak Pengujian (cm)	Waktu tahap 1 (detik)	Waktu tahap 2 (detik)	Waktu tahap 3 (detik)	Waktu Total (detik)	Penyambungan Konektor (berhasil/gagal)
1	450	8,91	45,94	1,85	56,7	Berhasil
2	450	8,11	42,35	2,08	52,54	Berhasil
3	450	12,91	38,9	2,14	53,95	Gagal
4	450	11,25	36,99	1,99	50,23	Berhasil
5	450	10,48	38,87	1,76	51,11	Gagal
6	450	12,76	39,47	1,87	54,1	Berhasil
7	450	10,3	48,72	1,72	60,74	Berhasil
8	450	9,9	40,18	1,55	51,63	Berhasil
9	450	9,42	39,22	1,63	50,27	Gagal
10	450	10,75	41,72	1,66	54,13	Berhasil
11	450	12,47	36,53	1,85	50,85	Berhasil
12	450	10,47	38,81	1,95	51,23	Berhasil
13	450	12,56	36,9	1,7	51,16	Berhasil
14	450	12,33	44,19	1,78	58,3	Berhasil
15	450	12,42	36,44	2,15	51,01	Gagal
16	450	12,37	38,89	1,79	53,05	Berhasil
17	450	12,93	43,65	1,72	58,3	Berhasil
18	450	9,27	39,58	1,56	50,41	Berhasil
19	450	12,69	40,73	1,65	55,07	Berhasil
20	450	10,84	40,4	1,55	52,79	Berhasil
21	450	9,89	38,23	1,68	49,8	Berhasil
22	450	8,61	49,73	1,61	59,95	Berhasil
23	450	12,17	39,54	2,15	53,86	Berhasil
24	450	9,71	48,87	1,98	60,56	Berhasil
25	450	8,29	42,55	2,13	52,97	Berhasil
Rata-Rata		10,87	41,09	1,82	**53,78**	Berhasil 21 kali

LAMPIRAN 2.B

Tabel pengujian auto docking dengan kontroler konvensional

No	Jarak Pengujian (cm)	Waktu tahap 1 (detik)	Waktu tahap 2 (detik)	Waktu tahap 3 (detik)	Waktu Total (detik)	Penyambungan Konektor (berhasil/gagal)
1	450	8,91	45,94	4,3	59,15	Berhasil
2	450	8,11	42,35	4,37	54,83	Berhasil
3	450	11,64	40,33	2,74	54,71	Berhasil
4	450	11,68	35,69	4,09	51,46	Berhasil
5	450	13,55	39,24	1,7	54,49	Gagal
6	450	11,94	39,34	1,65	52,93	Berhasil
7	450	12,14	41,87	2,51	56,52	Berhasil
8	450	11,78	54,55	2,52	68,85	Berhasil
9	450	11,53	52,12	1,97	65,62	Berhasil
10	450	10,62	41,62	1,88	54,12	Berhasil
11	450	14,52	48,56	2,09	65,17	Gagal
12	450	12,87	37,39	4,11	54,37	Berhasil
13	450	12,87	42,11	1,9	56,88	Berhasil
14	450	13,69	51,49	4,38	69,56	Berhasil
15	450	13,15	39,4	3,89	56,44	Gagal
16	450	13,18	45,69	2,83	61,7	Berhasil
17	450	11,18	53,81	4,18	69,17	Gagal
18	450	12,15	39,14	3,58	54,87	Berhasil
19	450	13,87	37,76	2,2	53,83	Berhasil
20	450	11,39	54,3	2,67	68,36	Berhasil
Rata-Rata		12,03	44,13	2,97	**59,15**	Berhasil 16 kali

DAFTAR PUSTAKA

[1] Kei HIROSE, Daisuke CHUGO, "*Camera-Based Localization for Indoor Service robots using Pictographs* ",2011 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM2011) Budapest, Hungary, July 3-7, 2011.

[2] Dimitris C. Dracopoulos," *Robot Path Planning for Maze Navigation* "1998 IEEE.

[3] K. Varun Raj et al "*A Beacon-based Docking System for an Autonomous Mobile Robot*" 13th National Conference on Mechanisms and Machines (NaCoMM07), IISc, Bangalore, India. December 12-13, 2007.

[4] G.Song, H.Wang, J. Zhang, T. Meng, " *Automatic Docking System for Recharging Home Surveillance Robots* " 13th IEEE Transactions on Consumer Electronics, Vol.57, No. 2, pp. 428-435. 2011.

[5] R. Cassinis, F. Tampalini, P. Bartolini, R. Fedrigotti, "*Docking And Charging System For Autonomous Mobile Robot*", Department Of Electronics For Automation, University Of Brescia Repository, Brescia. 2005.

[6] Ren C. Luo et al, "*Automatic Docking and Recharging System for Autonomous Security Robot*", Intelligent Robots and Systems. pp. 2953-2958. 2005.

[7] Hendawan soebhakti, "*Pengembangan Sistem Navigasi Mobile Robot Berdasarkan ekstraksi Ciri Lingkungan Koridor Gedung Menggunakan Sensor RGB-DEPTH*", Thesis program magister institut teknologi sepuluh nopember Surabaya 2012.

[8] T. Jyh-Hwa, L. Su Kuo " *The Development of the Restaurant Service Mobile Robot with a Laser Positioning System* " 27th Chinese Control Conference pp. 662-666. 2008.

[9] C. Hung Chen, A. Liu, Z. Pei-chuan " *Controlling a Service robot in a Smart Home with Behavior Planning and Learning* " IEEE International Conference on Systems, Man, and Cybernetics pp. 2821-2826. 2014.

[10] C. Chen, Q. Gao, Z. Song, O.Liping, X. Wu " *Catering Service robot* " Proceedings of the 8th World Congress on Intelligent Control and Automation, pp. 599-604. 2010.

[11] 11 Muhammad Fuad, " *Navigasi Berbasis kamera RGB-D untuk Robot Pembersih Lantai Dalam Koridor Pusat Robotika ITS* ", Thesis program magister institut teknologi sepuluh nopember surabaya 2012.

[12] W. Yi-Cheng, Ming-Chang, T Yi-Jeng " *Robot Docking station for Automatic Battery Exchanging and Charging* " Proceedings of the 2008 IEEE International Conference on Robotics and Biomimetics, pp. 1043-1046. 2010.

[13] D. Nadir Nourain, S. Brahim Belhaouari, J. Josefina ,"*Fast Template matching Method Based Optimized Sum of Absolute Difference Algorithm for Face Localization*", International journal of computer application, vol. 18, 30-34. 2011.

[14] Hui. Dong, Yuan. Han Dian, "*Research of Image Matching Algorithm Based on SURF Features*", 2012 International Conference on Computer Science and Information Processing (CSIP), 1140-1143. 2012.

[15] Guoying. Zhao, Timo. Ahonen, Jiri. Matas, Matti. Pietikainen, " *Rotation-Invariant And Video Description With Local Binary Patterns Features* ", IEEE transactions on image proccesssing, Vol. 21 No.4, pp. 1465-1477, 2012.

[16] H. Rami, M. Hamri, Lh. Masmoudi ," *Object tracking in image sequence using local binary pattern (LBP)*", International journal of computer application, vol. 63, 20-23. 2013.

[17] L. Kwon, L. Chulhee ," *Fast Object Detection Based on Color Histograms and Local Binary Patterns*", Korea Communications Agency Research KCA-2012-11913-04002. 2012.

[18] *Datasheet* Panasonic LC-R127R2.

[19] *Datasheet* driver motor AXHD50K.

RIWAYAT HIDUP PENULIS

Riza Agung Firmansyah dilahirkan di Bojonegoro, 04 Maret 1991. Merupakan anak ketiga dari tiga bersaudara pasangan Bapak Sukartono dan Ibu Sini. Penulis memulai pendidikan di SDN Ngraho I Kalitidu Bojonegoro. Lalu melanjutkan di SMPN 1 Padangan Bojonegoro. Penulis menempuh jenjang pendidikan selanjutnya di SMK MIGAS Cepu Blora. Setelah lulus pendidikan SMK pada tahun 2008 penulis melanjutkan pendidikan di Politeknik Elektronika Negeri Surabaya (PENS). Penulis mengambil jurusan D4 Teknik Elektronika dan menyelesaikan studi pada tahun 2012. Selanjutnya penulis meneruskan pendidikan di Program Magister Jurusan Teknik Elektro Institut Teknologi Sepuluh Nopember dengan memilih bidang keahlian Elektronika.

Email : agoenk.fir@gmail.com , riza.agung12@mhs.ee.its.ac.id

Halaman ini sengaja dikosongkan